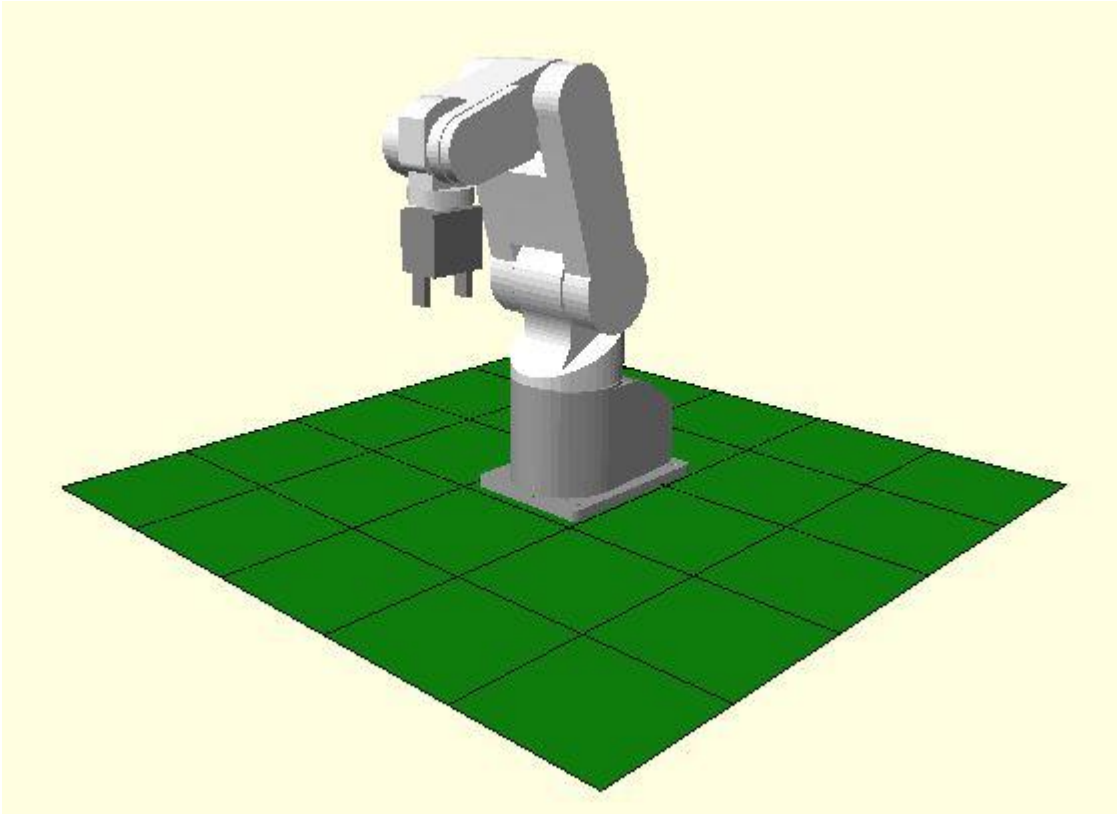
ÍNDICE ANEXOS

ANEXO A

A1. PRÁCTICA 1

Automatización y Robótica Industrial



PRÁCTICA 1

Instrucciones básicas

A1.1. Cuestiones previas

PRÁCTICA 1: CUESTIONES PREVIAS	**Fecha:**
Alumno:	**Grupo:**

Previamente a la realización de las cuestiones previas, deben comprenderse los conceptos explicados en el punto 10 de la memoria sobre Cosimir y el punto 11.4 que explica las instrucciones básicas del lenguaje Melfa Basic IV. Una vez se han entendido estos conceptos, se pueden observar algunos de los vídeos que se adjuntan en el punto 8.1 sobre Pick and Place, para poder comprender el sentido de la práctica. Esta información está disponible en el volumen I. Por último, se adjunta un tutorial de sobre cómo realizar la práctica 1:

https://www.youtube.com/watch?v=G-3wZmZrObw

Una vez se han realizado lo indicado anteriormente, se pueden realizar las cuestiones previas que se muestran a continuación:

1. Si se quiere conseguir un movimiento de un punto a otro a máxima velocidad, ¿qué tipo de instrucción debe utilizarse? (1 punto)
2. ¿Mediante la instrucción MOV se puede intuir la trayectoria que seguirá el robot? ¿Y mediante la instrucción MVS? Razona la respuesta explicando el por qué. (1,5 puntos)
3. ¿En qué momentos es necesario utilizar la instrucción MVS? ¿Por qué? (1 punto)
4. Si únicamente se tiene un utillaje en el robot, ¿es necesario numerar la herramienta? (1 punto)
5. ¿En qué ocasiones es más utilizada la instrucción DLY? ¿Por qué? (1 punto)

6. ¿Qué instrucción permite determinar la velocidad de los movimientos en %? ¿Y cuál en mm/s? ¿Cuál de ellas es más apropiada para definir la velocidad de un movimiento lineal? (1 punto)
7. ¿Es necesario definir la velocidad para cada punto? (1 punto)
8. ¿Qué sistema de coordenadas es útil mover cuando se graban los puntos para la recogida y dejada de la pieza? (1 punto)
9. ¿Es posible modificar una de las coordenadas de un punto en el código de programación? En el caso de que sea posible, pon un ejemplo de una posición cualquiera y posteriormente, como se realizaría la modificación y finalmente las coordenadas finales. (1,5 puntos)

A1.2. Desarrollo de la práctica

En el ejercicio que se presenta a continuación, se pondrán en práctica las instrucciones básicas de control de movimiento del robot. Para ello, se desea programar una tarea de *Pick and Place*, en la que el brazo industrial recoja una pieza en un punto estático sobre una caja y la deposite sobre otra caja.

El robot debe partir de una posición inicial o de reposo P1 y moverse hacia una posición de aproximación a 50 mm por encima de la recogida de la pieza. Posteriormente, debe bajar hasta P2 a 70 mm/s, coger la pieza y realizar el retorno de la misma manera. A continuación, volviendo a una velocidad de 100 mm/s, debe pasar por el punto inicial y moverse hacia una posición de aproximación a 50 mm por encima de la dejada de la pieza, y realizar un movimiento lineal de nuevo a 70 mm/sal hasta P5. Aquí dejaremos la pieza y subiremos linealmente 50 mm. Finalmente, el robot completará el ciclo regresando a P1.

La siguiente imagen representa gráficamente el proceso a realizar:

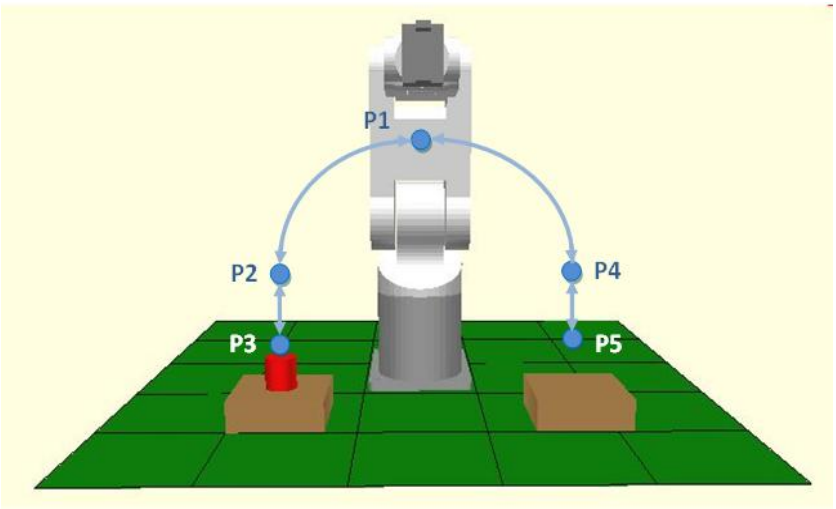


Figura A1.1. Movimientos básicos a realizar por el robot industrial realizando una tarea de *Pick and Place* (Fuente: Propia).

El primer paso que se debe hacer por tal de realizar una simulación de un proceso robotizado es crear el *layout*, es decir, el entorno con el que el robot trabajará.

A1.2.1. Creación del *layout*

En nuestro primer caso, se creará un *layout* muy sencillo, con solo dos cajas que indicarán las posiciones de recogida y dejada de la pieza y un cilindro que representará la pieza que el robot deberá transportar.

Primeramente, se crearán las dos cajas. Para ello, se debe seleccionar el icono de *Model Libraries* que se encuentra en la barra de herramientas. Una vez seleccionado, aparecerá la siguiente ventana:

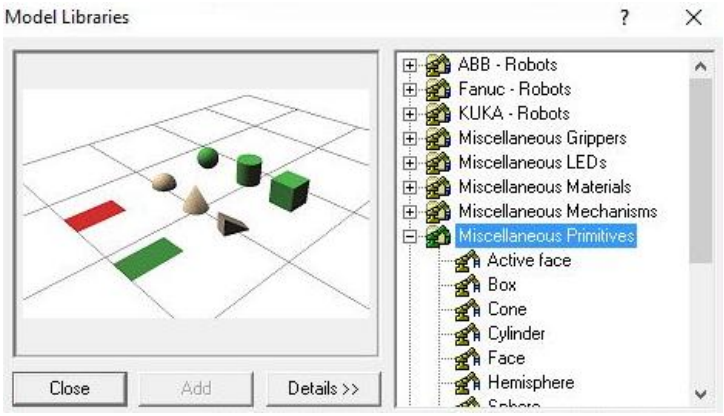


Figura A1.2. Creación de las cajas pertenecientes al *layout* en la ventana *Model Libraries* (Fuente: Propia).

En ella, se debe seleccionar la opción de *Miscellaneous Primitives*, donde se encuentran diferentes elementos. En este caso, se escoge la opción Box.

En el momento que se escoja una de las opciones, aparece la fotografía del elemento en la parte izquierda de la ventana y se habilitará el botón *Add*. Cuando se clique en el botón de añadir, se verá que la pieza ha sido añadida en la ventana de animación. Como en nuestro caso se tienen dos cajas, se podrá repetir el proceso clicando de nuevo el botón *Add*, para añadir la segunda caja.

Una vez se han insertado las cajas, se debe definir su posición y medidas. Para ello, se selecciona el botón de *Model Explorer On/Off*. En esta ventana, aparecerán todos los elementos que incluyen el *layout*.

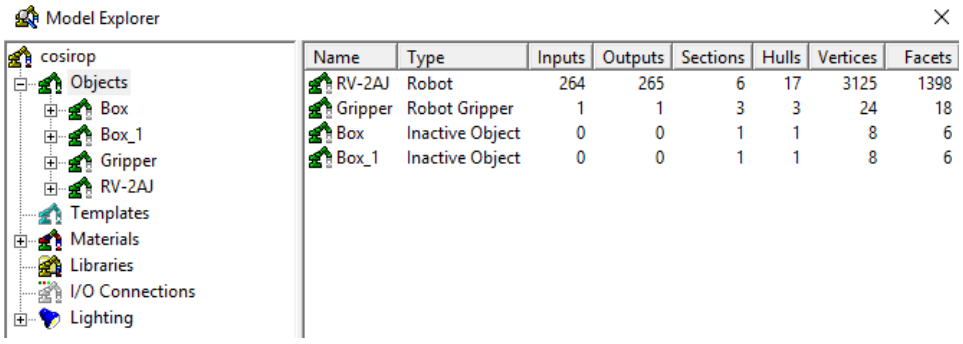


Figura A1.3. Componentes existentes en el *layout* actual (Fuente: Propia).

Por tal de definir la posición y medidas de las cajas, se debe seleccionar el elemento deseado, en nuestro caso *Box*, y desplegar el árbol en el que aparecerá la pestaña Base. Clicando sobre Base, en la parte derecha de la ventana, aparecerá la opción *Box 1*. Es aquí donde se debe clicar con el botón derecho, y aparecerá el listado de opciones siguiente:

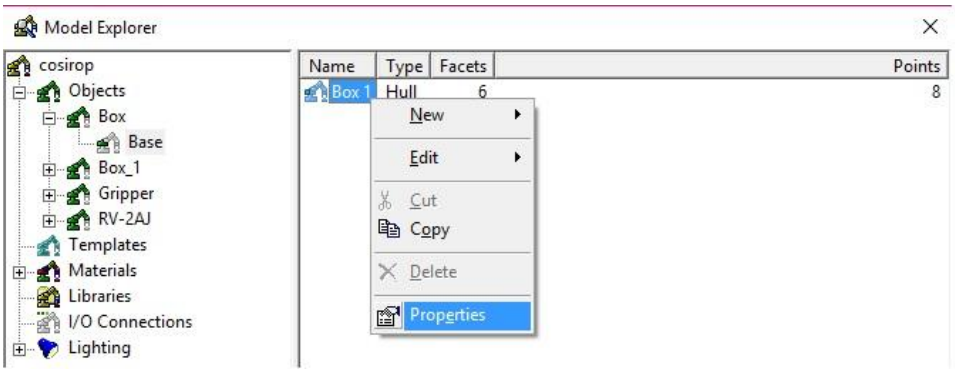


Figura A1.4. Proceso para la definición de las características de los elementos (Fuente: Propia).

Se debe seleccionar la opción *Properties*, y a continuación, aparecerá la siguiente ventana:

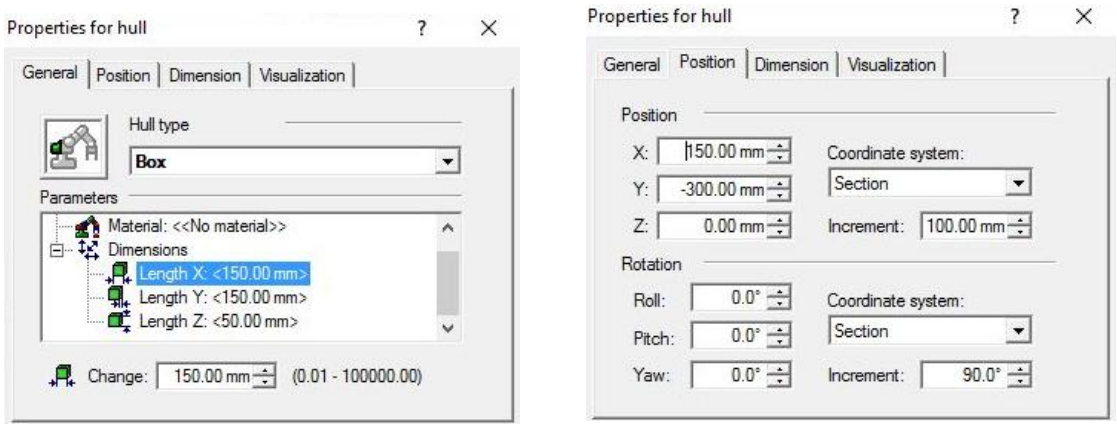


Figura A1.5. Definición de las propiedades de la primera caja: medidas y posición (Fuente: Propia).

En la pestaña *General*, se definirán los parámetros de medida de la caja. Por otro lado, en la pestaña *Position*, se definirá la posición de la caja. En el caso de que se quiera variar el color, se puede modificar en la pestaña *Visualization*. En nuestro caso, se ha variado a un color marrón.

Para la configuración de la segunda caja, se debe repetir el proceso, estableciendo las mismas medidas, únicamente cambiando su posición:

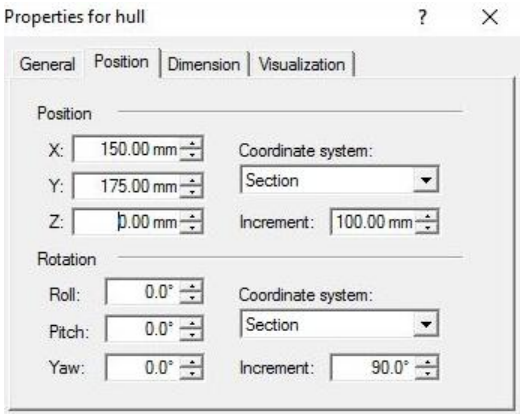


Figura A1.6. Definición de la posición de la segunda caja (Fuente: Propia).

A continuación, se definirá lo que será la pieza que manipulará el robot industrial. En este caso, se escogerá un cilindro. Deberá repetirse el proceso, seleccionando esta geometría en la pestaña de *Model Libraries* y seguidamente desplegar la opción de *Miscellaneous Primitives* donde se encuentra el cilindro. Se debe seleccionar y posteriormente clicar *Add*, como en el caso de la caja.

Una vez se ha introducido el elemento en el entorno de trabajo, se deben definir sus propiedades. Para ello se debe seleccionar la opción *Model Explorer On/Off,* donde se encuentra, además de las cajas ya definidas, el cilindro.

Se debe desplegar el árbol dentro de *Cylinder*, y donde aparece *Cylinder 1*, clicar con el botón derecho y en el listado seleccionar *Properties*.

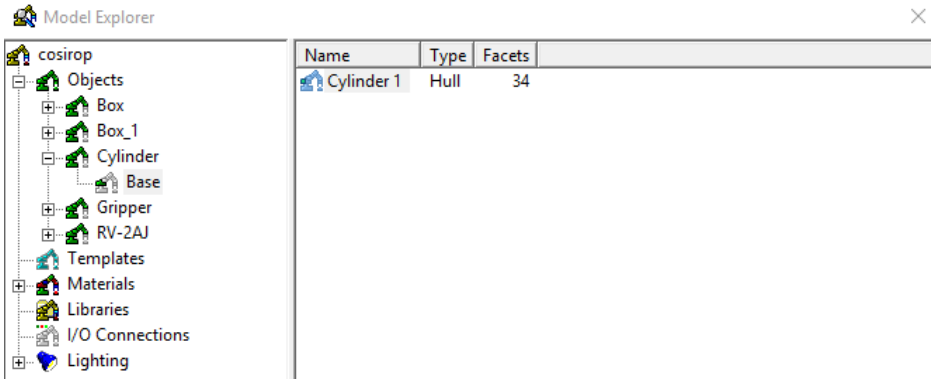


Figura A1.7. Paso para la selección de las propiedades del cilindro (Fuente: Propia).

Aquí se encontrará de nuevo (mismo proceso que con las cajas) la ventana para definir las medidas y posiciones del cilindro.

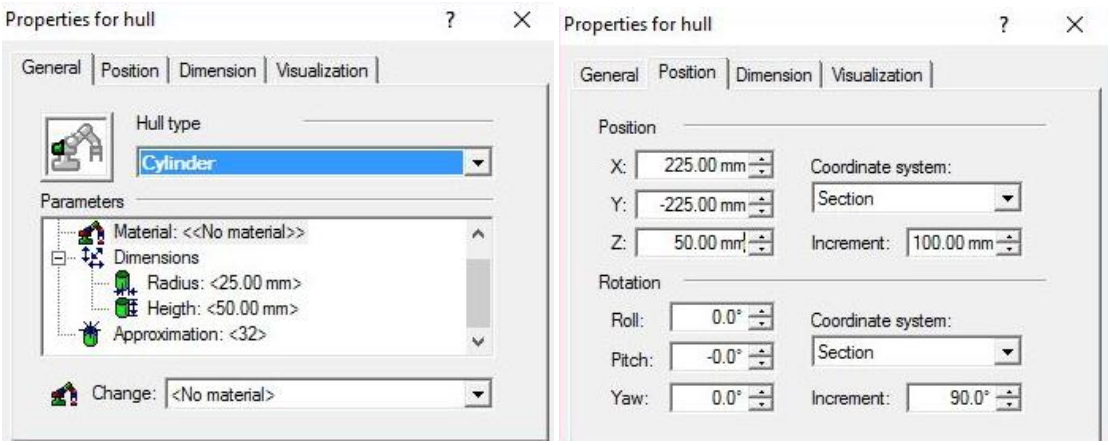


Figura A1.8. Definición de las propiedades del cilindro: medidas y posición (Fuente: Propia).

En este caso, como el cilindro se trata de la pieza que será manipulada por el robot, se necesita definir un *Grip Point*, es decir, un punto que hará que el robot pueda realizar el agarre de la pieza mediante la garra o *gripper*.

Con la finalidad de definir este punto, se debe clicar con el botón derecho en Base, dentro de *Cylinder*, posicionar el mouse sobre *New*, y, por último, seleccionar *Grip Point*.

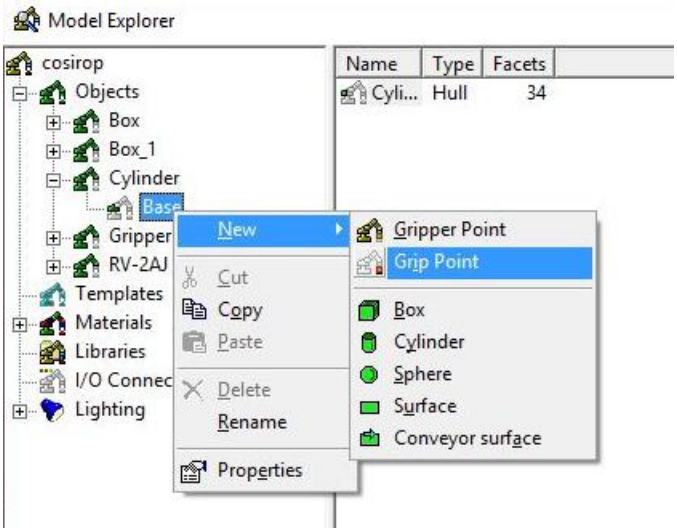


Figura A1.9. Creación del *Grip Point* (Fuente: Propia).

A continuación, aparecerá el *Grip Point* creado en la parte derecha de la ventana *Model Explorer*. Se debe clicar con el botón derecho sobre él y seleccionar *Properties*.

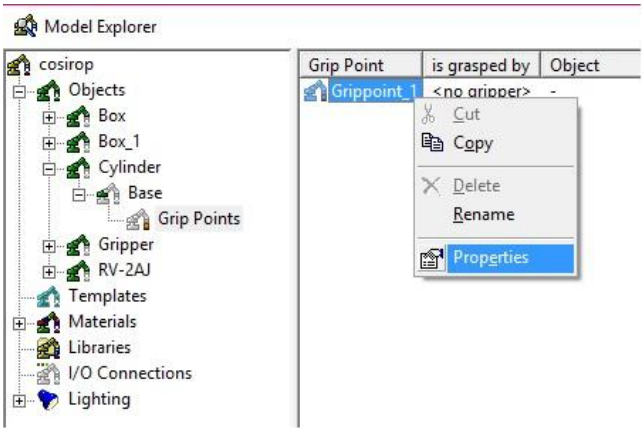


Figura A1.10. Selección de las propiedades del *Grip Point* (Fuente: Propia).

Como en los casos anteriores, se debe definir una posición de *Grip Point*. La posición en X e Y ha de ser la misma que la de la pieza que cogerá el robot en el proceso.

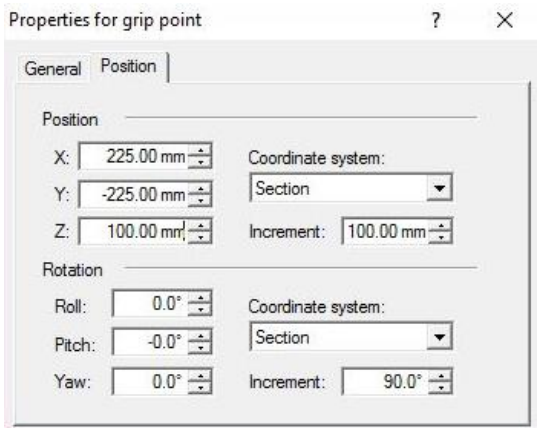


Figura A1.11. Configuración de la posición de *Grip Point* (Fuente: Propia).

A1.2.2. Grabación de los puntos

Para la correcta realización de la práctica, previamente a la programación, se deben tener claros los puntos clave por los que deberá moverse el robot. En el caso de este enunciado serán 5 puntos:

- **P1**: Posición inicial o de reposo.
- **P2**: Posición de aproximación para recogida de la pieza.
- **P3**: Posición para coger la pieza.
- **P4**: Posición de aproximación para dejada de la pieza.
- **P5**: Posición para dejar la pieza.

Para poder posicionar el robot en los puntos deseados, se debe abrir la ventana *Jog Operation*, que se puede encontrar dentro del menú *Extras*, en *Teach-In*.

La estructura de la ventana se observa en la siguiente fotografía:

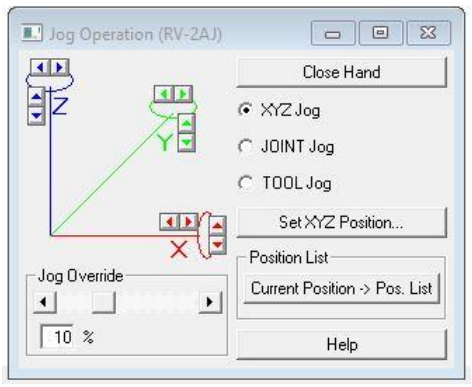


Figura A1.12. Grabación de los puntos (Fuente: Propia).

Se pueden diferenciar 3 apartados distintos según el eje de coordenadas con el que deseamos mover el robot en el entorno:

- **XYZ Jog**: El robot se mueve respecto al sistema de coordenadas base.

- **JOINT Jog**: Se pueden mover cada uno de los ejes del robot individualmente. La posición, únicamente en este caso, se dará en grados.

- **TOOL Jog**: El robot se mueve respecto al sistema de coordenadas herramienta.

Mediante las flechas que aparecen en el dibujo, se puede desplazar o rotar el robot en un eje en concreto. Por otro lado, el *Jog Override*, permite variar la velocidad a la que se realiza el desplazamiento.

Por tal de establecer unas coordenadas que vengan dadas, se puede utilizar la opción *Set XYZ Position*, e introducir los datos. Una vez se tenga el robot en la posición deseada, podemos clicar *Current Position -> Pos. List*, y la posición será grabada en la lista de puntos.

En nuestro caso, utilizando el sistema de coordenadas XYZ, y mediante la opción *Set XYZ Position*, se introducen las siguientes coordenadas:

No	Position	Orientation	Comment
P5	225.0, 250.0, 75.0	0, 180,R,A	Dejada pieza
P4	225.0, 250.0, 125.0	0, 180,R,A	Posición de aproximación
P3	225.0,-225.0, 75.0	-0, 180,R,A	Recogida pieza
P2	225.0,-225.0, 125.0	-0, 180,R,A	Posición de aproximación
P1	250.0, 0.0, 300.0	-0, 180,R,A	Posición inicial

Figura A1.13. Listado de puntos grabados (Fuente: Propia).

Una vez han sido creados los puntos, se podrá llevar a cabo la programación de ellos, por tal de que el robot realice el proceso planteado.

A1.2.3. Programación del proceso

Para ello, se deben tener claras las instrucciones que se necesitarán utilizar en la programación. En este caso, se utilizarán las siguientes:

- **MOV**: Realiza una interpolación en los ejes del robot punto a punto, para realizar el movimiento más sencillo y rápido. Recordad que con esta instrucción no se puede intuir la trayectoria.

- **MVS**: Realiza una interpolación lineal, con lo cual obtenemos como resultado una trayectoria lineal.

- **OVRD**: Define el porcentaje de velocidad máxima

- **HOPEN/HCLOSE**: Abrir/Cerrar pinza.

- **DLY**: Retardo de unos segundos.

Observando la fotografía del enunciado, donde se indican los puntos que se deben crear, es importante tener en cuenta que los puntos que rodean la instrucción de coger y dejar pieza deben realizarse con un movimiento lineal, el resto pueden realizarse mediante la instrucción MOV. Es importante destacar, que hace falta añadir retardos que simulen los tiempos de apertura y cierre de la pinza antes de iniciar movimientos posteriores.

El programa final quedaría de la siguiente manera:

```
'-----------------------------------------------------------
'           PRÁCTICA 1 - INSTRUCCIONES BÁSICAS
'   Autor: Sandra Corchado
'   Fecha: Octubre 2017
'-----------------------------------------------------------

10 HOPEN 1          'Abrir Pinza
11 OVRD 100         'Velocidad al 100% de la velocidad total

12 MOV P1           'Movimiento hacia la posición inicial
13 MOV P2           'Movimiento hacia la posición de aproximación de recogida de la pieza
14 SPD 70           'Velocidad a 70 mm/s
15 MVS P3           'Movimiento lineal para recogida de la pieza
16 HCLOSE 1         'Cerrar Pinza
17 DLY 1            'Retardo de 1 s
18 MVS P2           'Movimiento hacia la posición de aproximación
19 SPD 100          'Velocidad a 100 mm/s
20 MOV P1           'Movimiento hacia la posición inicial
21 MOV P4           'Movimiento hacia la posición de aproximación de dejada de la pieza
22 SPD 70           'Velocidad a 70 mm/s
23 MVS P5           'Movimiento lineal para dejada de la pieza
24 HOPEN 1          'Abrir Pinza
25 DLY 1            'Retardo de 1 s
26 MVS P4           'Movimiento hacia la posición de aproximación
27 SPD 100          'Velocidad a 100 mm/s
28 MOV P1           'Movimiento hacia la posición inicial
29 END              'Fin del programa
```

A1.2.4. Compilación del proyecto, compilación de programa y ejecución

Antes de compilar y ejecutar el programa, se debe clicar sobre *Project Management*, situado dentro del menú *Extras* o *Open Project Management* en uno de los iconos de la barra de herramientas. Aparecerá la siguiente ventana donde estarán incluidos los archivos .POS (de posición) y .MBA (código de programa):

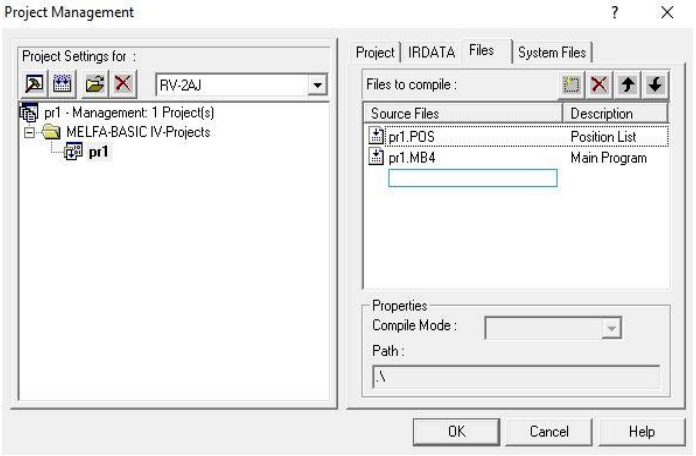


Figura A1.14. Añadir un proyecto (Fuente: Propia).

Para hacer activo este proyecto y compilarlo para comprobar que no hay errores, se debe clicar con el botón derecho sobre el proyecto creado, y seleccionar *Set as Active Project* y seguidamente, *Compile pr1-Project*. Finalmente, pulsar OK.

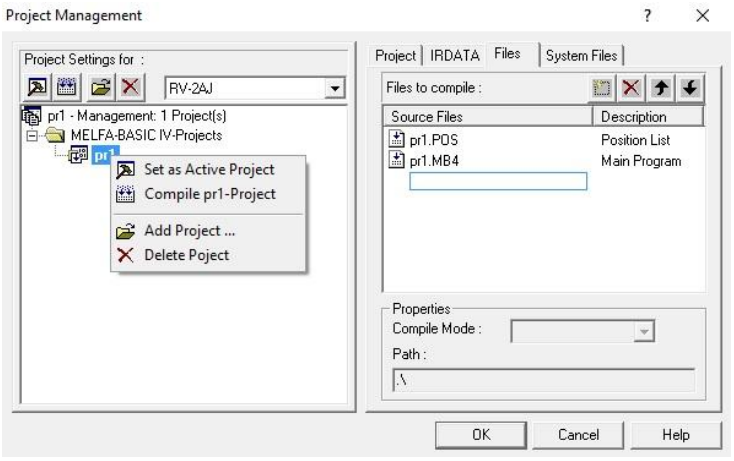


Figura A1.15. Creación, activación y compilación del proyecto (Fuente: Propia).

Una vez compilado, en la ventana de mensajes de la parte inferior de Cosimir®, debe aparecer que no se han encontrado errores.

Posteriormente, se puede compilar y cargar el programa al robot, y de nuevo comprobar que en la ventana de mensajes no ha aparecido ningún error. En el caso de que aparezca, se dispondrá de una pequeña descripción de donde se encuentra el error.

Finalmente, se inicia la ejecución del programa de manera que se mostrarán los movimientos programados en la ventana de animación.

Ampliaciones:

1) En la programación, es aconsejable definir el mínimo número de puntos posibles, ya que cuando tenemos un programa muy extenso, es más fácil identificar las acciones o posibles fallos cuanto menor sean los puntos a comprobar. Para ello, se han de eliminar los puntos de aproximación definidos anteriormente: P2 y P4 y utilizar una de las funciones de las instrucciones de movimiento:

MOV P3, -50: Realizará un movimiento hasta una posición 50 mm por encima de la posición P3.

MVS P3, -50: Realizará un movimiento lineal de 50 mm hacia arriba una vez recogida la pieza.

El mismo proceso tendrá lugar con el punto P5.

2) Para poder practicar el movimiento de ejes del robot, la creación de puntos y de instrucciones de movimiento, modifica la posición de dejada, de manera que el cilindro quede colocado de manera horizontal sobre la mesa.

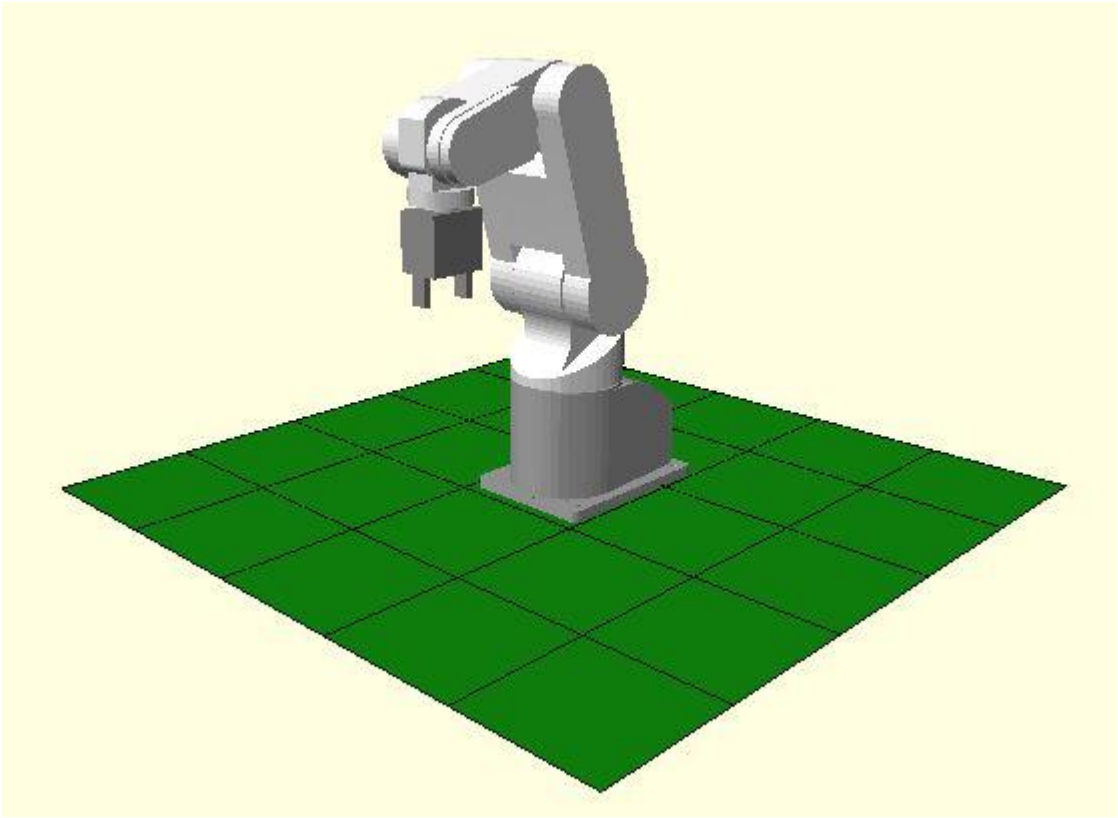
A1.3. Cuestiones finales

PRÁCTICA 1: CUESTIONES FINALES	**Fecha:**
Alumno:	**Grupo:**

1. ¿Qué alternativa se tiene para programar un punto previo que no sea MOV PX,-50? (1,5 puntos)
2. ¿Por qué es necesario minimizar la velocidad en los puntos de recogida y dejada de la pieza? (1 punto)
3. ¿Se podría haber realizado todo el programa mediante movimientos continuos por tal de aumentar la velocidad de realización del proceso? Justifica tu respuesta (1 puntos)
4. Si cuando se resetea el programa la pinza ya aparece abierta, ¿por qué es necesario introducir HOPEN 1 en la primera línea de programa? (1,5 puntos)
5. En la ampliación 2, ¿qué eje de coordenadas es más conveniente mover para la creación de la nueva posición de dejada? (1 punto)
6. Pon un ejemplo de aplicación industrial en el que se realice un Pick and Place y indica con qué modelo de robot se realiza esta aplicación o cuál es el modelo más conveniente. (2 puntos)
7. Redacta las conclusiones y conceptos aprendidos en la práctica 1. (2 puntos)

A2. PRÁCTICA 2

Automatización y Robótica Industrial



PRÁCTICA 2

Instrucciones de movimiento

A2.1 Cuestiones previas

PRÁCTICA 2: CUESTIONES PREVIAS	**Fecha:**
Alumno:	**Grupo:**

Previamente a la realización de las cuestiones previas, deben comprenderse los conceptos explicados en el punto 11.4 que explica las instrucciones de movimiento del lenguaje Melfa Basic IV, como son los movimientos circulares. Una vez se han entendido estos conceptos, se pueden observar algunos de los vídeos que se adjuntan en el punto 8.4 sobre como los robots aplican cola o adhesivo sobre otros materiales, para poder comprender el sentido de la práctica. Esta información está disponible en el volumen I. Por último, se adjunta un tutorial de sobre cómo realizar la práctica 2:

https://www.youtube.com/watch?v=TJY1djzbGC8&feature=youtu.be

Una vez se ha visto el tutorial adjuntado, se puede pasar a la resolución de las cuestiones previas:

1. ¿Qué instrucción permite realizar un arco mediante un punto de referencia? Pon un ejemplo de en qué situaciones podría ser útil este tipo de instrucción. (1 punto)
2. ¿Qué tipo de instrucción permite realizar un círculo completo? Pon un ejemplo de en qué situación se podría utilizar esta instrucción. (1 punto)
3. ¿Qué orden de puntos debe seguir la instrucción MVR? (1 punto)
4. ¿Cómo determinarías que instrucción es más adecuada para la creación de un arco? (1 punto)
5. ¿Podría una instrucción MOV simular un arco grabando solo 3 puntos como para las instrucciones anteriores? ¿Por qué? (1,5 puntos)

6. ¿Puede utilizarse un mismo punto como final de un arco y el punto inicial del siguiente? (1 punto)
7. ¿Qué instrucción es necesaria para disminuir la velocidad en los movimientos circulares? (1 punto)
8. ¿Existe la posibilidad de visualizar la trayectoria que sigue el robot? (1 punto)
9. ¿Qué problema puede causar al robot programar un círculo completo sin modificar orientaciones? (1,5 puntos)

A2.2. Desarrollo de la práctica

La segunda práctica, tiene el propósito de poner en práctica las instrucciones para realizar movimientos circulares. Para ello, hemos escogido una aplicación típica en el mundo industrial como puede ser la introducción de adhesivo/cola en piezas. En nuestro caso, el adhesivo se deberá colocar en el contorno de la puerta de un coche, para después poder colocar la goma sobre este adhesivo.

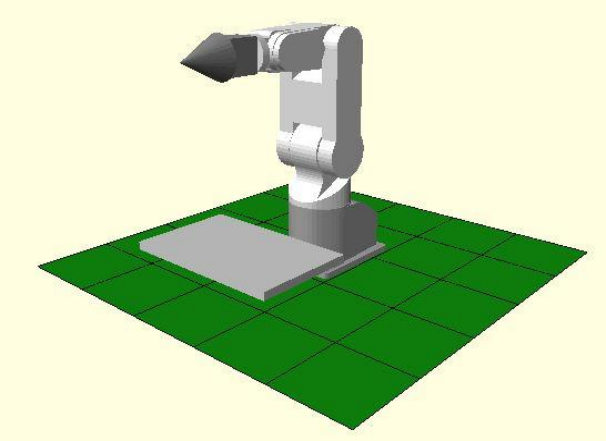


Figura A2.1. *Layout* creado para la creación de movimientos circulares (Fuente: Propia).

Con el fin de realizar este proceso, primeramente, se ha seleccionado una fotografía de la puerta de un coche para tener una idea del contorno habitual. Cabe destacar que la cola se aplica por el lado contrario, es por lo que en la simulación los puntos serán grabados por el que sería el lado correspondiente. Una idea aproximada de los puntos que deben crearse se muestra en la fotografía siguiente:

Figura A2.2. Puntos que se deben crear para seguir el contorno de la puerta de un coche (Fuente: [7])

Como se puede observar en la fotografía A2.2, se partirá del extremo superior derecho, posición P1, hasta llegar a la posición inferior P2 creando un arco que pasa por P3. A partir de P2, debe definirse un movimiento lineal que llegue hasta la posición P4, donde tendrá lugar otro arco formado por los puntos P4 > P6 > P5. Donde termina el arco anterior, empieza un siguiente arco más grande formado por P5 > P8 > P7. Este arco, irá seguido de nuevo por otro arco de tamaño inferior, formado por los puntos P7 > P10 > P9. Por último, para crear el arco superior, únicamente se debe crear otro punto más, ya que estará formado por la posición P10, como punto inicial, P1 como punto final y por tanto, P11 será la posición intermedia que faltará por crear.

Los movimientos de aplicación del adhesivo deben realizarse a una velocidad más reducida, 70 mm/s.

A2.2.1. Creación del *layout*

Previamente a la simulación, se deben introducir los elementos necesarios para realizar el proceso. Para ello clicaremos en el icono de *Model Libraries* disponible en la barra de herramientas, y desplegando la pestaña *Miscellaneous Primitives* se encuentran todos los elementos necesarios para esta práctica.

Será necesario introducir una caja que actuará como puerta, ya que en Cosimir® no se dispone de elementos con formas, y el cono que simulará la herramienta.

Una vez introducidos los elementos, se debe configurar el tamaño y posición. Para ello, es necesario clicar sobre el icono *Model Explorer On/Off* y aquí se encontrarán todos los elementos que forman el *layout* actualmente, tal y como se muestra en la imagen siguiente:

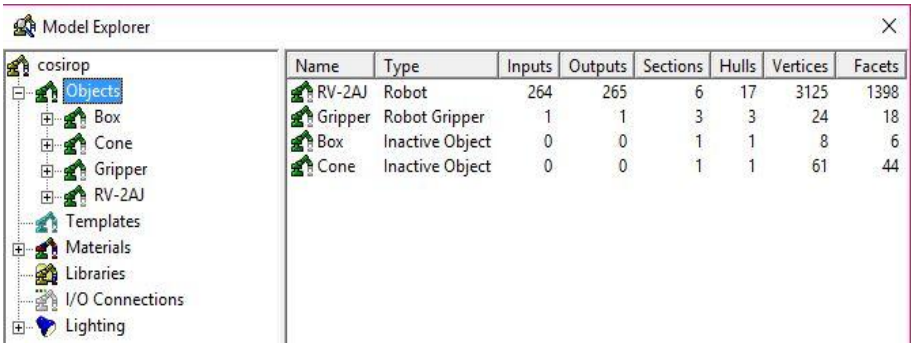


Figura A2.3. Componentes existentes en el *layout* actual (Fuente: Propia).

Como se explicó en la práctica anterior, se debe desplegar el símbolo + de cada componente creado, y aparecerá en cada uno de ellos la opción Base. Clicando sobre Base, en la parte derecha de la ventana aparecerá una nueva opción, que para todos los casos se deberá clicar con el botón derecho y seleccionar propiedades.

En esta ventana es donde se deben hacer los cambios necesarios de tamaño (pestaña *General*), posición (pestaña *Position*), y el color en el caso de querer hacer el entorno del robot más visible (pestaña *Visualization*).

Para el caso de la caja, serán modificados tanto la posición como las dimensiones. Para el caso del cono, el programa no da opción a realizar un cambio en el tamaño, por tanto, únicamente se especifica la posición. Por otro lado, al tratarse de la pieza que debe ser manipulada por el robot, hay que crear y definir un *Grip Point*. La posición del *Grip Point* se corresponde con la posición del cono. Los datos de interés sobre las características de los componentes se muestran en la tabla siguiente:

Tabla A2.1. Posiciones de los componentes del *layout* (Fuente: Propia).

	Dimensiones			**Posición**		
	X (mm)	**Y (mm)**	**Z (mm)**	**X (mm)**	**Y (mm)**	**X (mm)**
Box	250	400	20	150	-200	100
Cone	-	-	-	310	0	550

A2.2.2. Grabación de los puntos

Una vez ha sido creado el entorno de trabajo del robot, se graban los puntos necesarios en el proceso. Se debe mover el robot a las posiciones necesarias mediante *Teach-In*, situado en el menú Extras, como se ha explicado anteriormente. Se introducen las coordenadas mediante la opción *Set XYZ Position* y cuando el robot se encuentre en la posición correcta, se clica sobre *Current Position -> Pos. List*. Las posiciones finales son las siguientes:

No	Position	Orientation	Comment
P11	345.0, 25.0, 170.0	-0, 180,R,A	Punto intermedio arco 5
P10	300.0,-115.0, 170.0	-0, 180,R,A	Punto intermedio arco 4
P9	315.0,-100.0, 170.0	0, 180,R,A	Punto final arco 4/Punto inicial arco 5
P8	255.0,-130.0, 170.0	0, 180,R,A	Punto intermedio arco 3
P7	275.0,-125.0, 170.0	0, 180,R,A	Punto final arco 3/Punto inicial arco 4
P6	220.0,-115.0, 170.0	0, 180,R,A	Punto intermedio arco 2
P5	235.0,-125.0, 170.0	0, 180,R,A	Punto final arco 2/Punto inicial arco 3
P4	215.0,-100.0, 170.0	-0, 180,R,A	Punto inicial arco 2
P3	275.0, 120.0, 170.0	-0, 180,R,A	Punto intermedio arco 1
P2	215.0, 135.0, 170.0	-0, 180,R,A	Punto final arco 1
P1	350.0, 135.0, 170.0	0, 180,R,A	Punto inicio arco 1/Punto final arco 5

Figura A2.4. Listado de puntos grabados (Fuente: Propia).

A2.2.3. Programación del proceso

Una vez definidas las posiciones, se han de tener claro las instrucciones a utilizar. En este caso, la instrucción más utilizada es la siguiente es MVR, en la que se escribe un arco pasando por 3 puntos.

Esta instrucción, como se explica en el punto 11.5.1, se ha de escribir de la siguiente manera: MVR Punto inicial, Punto intermedio o de tránsito, Punto final

El programa final, debería estar estructurado de la siguiente manera:

```
'------------------------------------------------------------
'            PRÁCTICA 2 - INSTRUCCIONES DE MOVIMIENTO
'    Autor: Sandra Corchado
'    Fecha: Octubre 2017
'------------------------------------------------------------

HCLOSE 1              'Cerrar pinza para coger herramienta
OVRD 100              'Velocidad al 100 %

MOV P1, -50           'Movimiento de aproximación al punto inicial
SPD 70                'Velocidad a 70 mm/s
MVS P1                'Movimiento lineal hacia el punto inicial
MVR P1, P3, P2        'Descripción del arco 1 pasando por los puntos P1, P3, P2
MVS P4                'Movimiento lineal hacia P4
MVR P4, P6, P5        'Descripción del arco 2 pasando por los puntos P4, P6, P5
MVR P5, P8, P7        'Descripción del arco 3 pasando por los puntos P5, P8, P7
MVR P7, P10, P9       'Descripción del arco 4 pasando por los puntos P7, P10, P9
MVR P9, P11, P1       'Descripción del arco 5 pasando por los puntos P9, P11, P1
MVS P1, -50           'Movimiento de aproximación al punto final
END
```

A2.2.4. Guardar proyecto, compilación y ejecución

Una vez programado el proceso y previamente a compilar el programa, se debe activar y compilar el proyecto. Para ello, es necesario seguir las indicaciones creadas en el punto A1.2.4 de la práctica anterior, ya que el proceso se realiza de la misma manera para cualquier proyecto.

Una vez se han establecido como proyecto activo el actual y compilado en la ventana de *Project Management*, se puede pasar a compilar el programa en el botón de la barra de herramientas con el nombre *Compile*. Al clicar este botón se debe observar la ventana de mensajes. En el caso de que aparezcan errores, hay que corregirlos previamente a cargar el programa en el robot. Una vez eliminados los errores, clicar sobre *Compile + Link*.

Finalmente se puede clicar *Start*, y observar el proceso programado en la ventana de animación.

Ampliaciones:

1) Modifica el programa substituyendo las instrucciones MVR por MVR2.

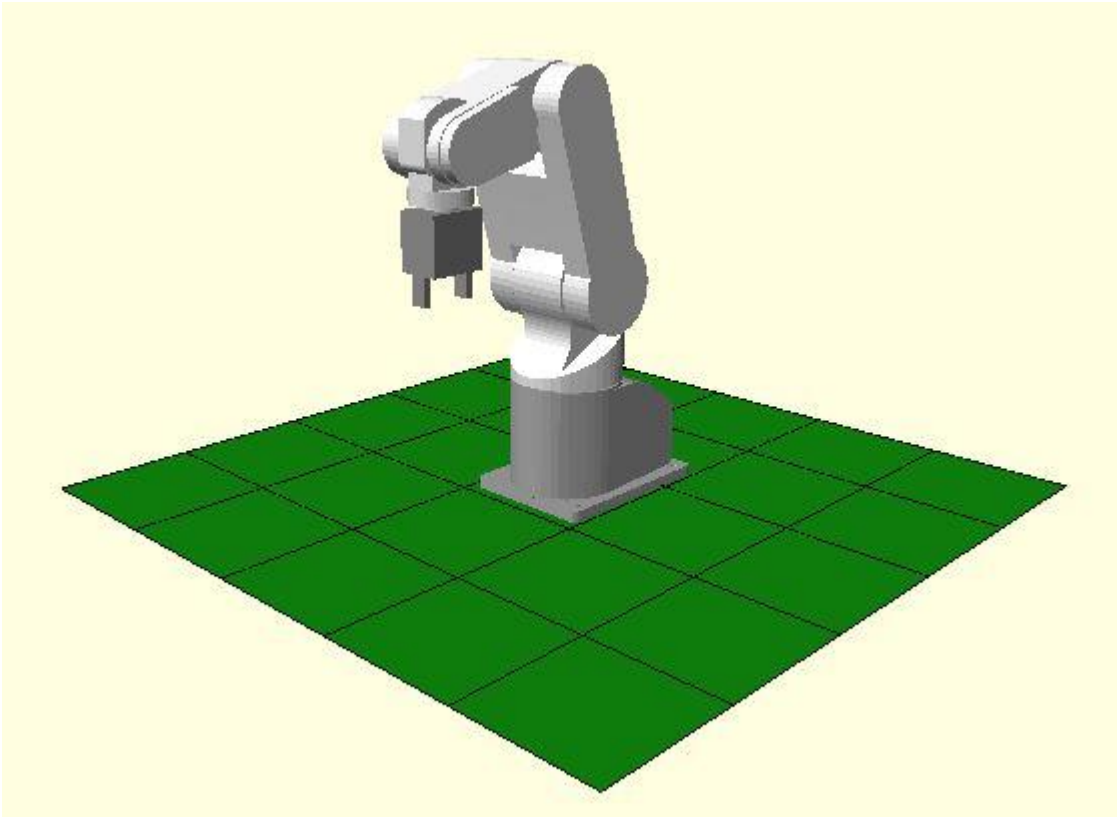
A2.3. Cuestiones finales

PRÁCTICA 2: CUESTIONES FINALES	**Fecha:**
Alumno:	**Grupo:**

1. ¿Es posible realizar alguna de las instrucciones con movimientos continuos? (1 punto)
2. ¿Qué inconvenientes presenta la opción TCP tracking que se observa en la práctica? (1 punto)
3. En el caso de encontrar un límite mecánico en alguno de los arcos, ¿qué modificaciones podrían realizarse para solucionar dicho problema? (2 puntos)
4. ¿Existe la posibilidad de programar 3 puntos a los que el robot alcance sin problemas y a la hora de realizar el movimiento circular no sea posible por la aparición de algún tipo de error? Justifica tu respuesta. (2 puntos)
5. Puede existir alguna manera de conseguir disminuir la cantidad de puntos grabados? (1 punto)
6. Pon un ejemplo de qué otra aplicación industrial podría ser realizada para reseguir un perfil mediante movimientos circulares. (1 punto)
7. Redacta las conclusiones y conceptos aprendidos en la práctica 2. (2 puntos)

A3. PRÁCTICA 3

Automatización y Robótica Industrial



PRÁCTICA 3

Funciones lógicas

A3.1. Cuestiones previas

PRÁCTICA 3: CUESTIONES PREVIAS	**Fecha:**
Alumno:	**Grupo:**

Previamente a la realización de las cuestiones previas, deben comprenderse los conceptos explicados en el punto 11.6 sobre los comandos de control del programa y el punto 11.7 donde se explica cómo realizar el control de las variables de entrada y salida (ambos disponibles en el volumen I). Para la ayuda a la realización de la práctica, se adjunta un tutorial donde se explican los pasos a realizar:

https://www.youtube.com/watch?v=oH_VTIkIzCg&feature=youtu.be

Una vez asumidos los conceptos indicados y se ha visto el vídeo adjuntado, se puede proceder a la realización de las cuestiones previas:

1. Numera las entradas y salidas que están reservadas para el controlador del robot RV-2AJ y indica la función que realiza cada una. (1 punto)
2. ¿Qué salidas del robot se encargan de gobernar la pinza? (0,5 puntos)
3. ¿Es posible imponer por código una activación de una señal de entrada? ¿Por qué? (1 punto)
4. ¿Pueden ser dos componentes conectados a una misma entrada/salida? (1 punto)
5. ¿Es posible asignar un valor a una variable en la misma línea donde se crea? (1 punto)
6. Si a una variable no se le asigna ningún valor, ¿por defecto es 0 o 1? (1 punto)

7. ¿En qué ocasiones puede ser útil utilizar la instrucción de espera? (0,5 puntos)
8. Explica la función de la instrucción condicional IF, y pon un ejemplo de un caso en el qué podría ser utilizada teniendo en cuenta dos señales: una de entrada y otra de salida. (1 punto)
9. Explica la función de la instrucción de bucle WHILE, y pon un ejemplo de un caso en el que podría ser utilizada. (1 punto)
10. ¿Existen diferencias entre la instrucción FOR y WHILE? (1 punto)
11. ¿En qué ocasiones es útil un Select/Case? (1 punto)

A3.2. Desarrollo de la práctica

La siguiente práctica consistirá en programar un proceso teniendo en cuenta señales externas de entrada y salida que pueden modificar los movimientos según la información que proporcionen.

Para ello, se creará un *layout* en el cual dispondremos de un *conveyor* que proporcione las piezas a recoger. Estas piezas serán detectadas mediante un sensor, que deberá enviar la información al robot industrial, para que éste pare la cinta y se disponga a recoger la pieza. Finalmente, las piezas serán dejadas en sus correspondientes mesas según el color de cada una.

Recordad que previamente a la posición de recogida y dejada de la pieza, el robot deberá pasar por un punto de aproximación a 50 mm y posteriormente el descenso a las posiciones P1, P2, P3 y P4, se realizarán con movimientos lineales a 70 mm/s.

Los componentes necesarios y un esquema de los movimientos del robot quedan resumidos en la siguiente figura:

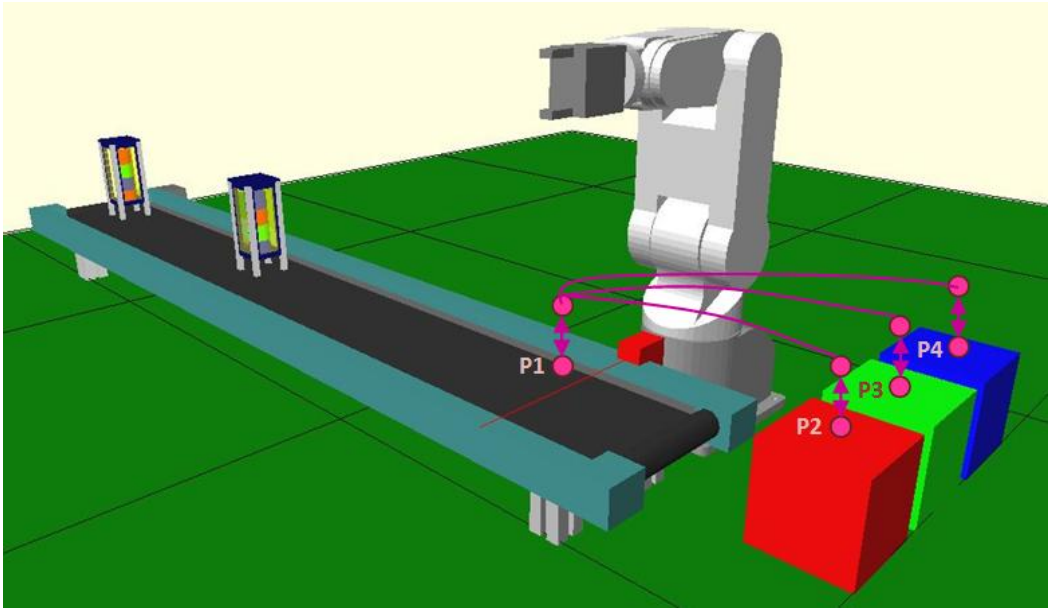


Figura A3.2. Creación de un *layout* para un *Pick and Place* combinado con entradas y salidas (Fuente: Propia).

A3.2.1. Creación del *layout*.

Con tal de crear el *layout* de la nueva simulación, se debe acceder a *Model Libraries* y añadir los componentes necesarios para el proyecto.

Primeramente, se necesitará la cinta por donde aparecerán los componentes. Se puede encontrar dentro de *Miscellaneous Mechanisms, Conveyor Belt 1*. En esta misma pestaña, se pueden añadir los creadores de los componentes, clicando *Parts Feeder 1* (añadir dos).

Por otro lado, se necesitará un sensor que diferencie los colores de las piezas que entren por la cinta (sensor RGB). Para ello se dispone de *Color Sensor*, dentro de la pestaña *Miscellaneous Sensor.*

Por último, en la pestaña *Miscellaneous Primitives*, se deberán añadir 4 *Box*, ya que tres de ellas actuarán como mesa para dejar los componentes, y una de ellas servirá como referencia para grabar los puntos.

Una vez creados todos los componentes del *layout*, deben ser posicionados correctamente. A continuación, se presenta una tabla con la posición para cada elemento:

Tabla A3.1. Posiciones de los componentes del *layout* (Fuente: Propia).

	X (mm)	**Y (mm)**	**Z (mm)**
ConveyorBelt	450	-1600	-350
SensorColor	160	0	155
Feeder1	290	-1400	140
Feeder1_1	290	-800	140
Box_1	-100	250	0
Box_2	80	250	0
Box_3	-280	250	0

El tamaño de las cajas que se ha utilizado como mesas de color para colocar las piezas, tienen unas dimensiones de 150x150x150 mm.

La última caja que se ha creado servirá como referencia para crear los puntos. Para ello, se posicionará la caja en el punto de recogida, es decir, el punto donde el sensor detecta la pieza. En nuestro caso, corresponderá a las coordenadas X 265 mm, Y -25 mm y Z 140 mm. También es necesario crear un *Grip Point* en X 290 mm, Y 0 mm y Z 140 mm, por tal de que el robot agarre la pieza y podamos grabar la posición de los puntos de dejada.

A3.2.2. Asignación de entradas y salidas del robot.

Previamente a la programación, es necesario conectar las diferentes señales de los elementos que forman el *layout* a las entradas y salidas que dispone el robot.

En el caso del sensor, será este mismo el que enviará una señal al robot, por tanto, las salidas del sensor deben ser conectadas a las entradas del robot. Cada salida del sensor (una para cada color), irá conectada a una entrada independiente del robot.

Por tal de realizar la conexión, únicamente se deben arrastrar una a una las salidas del sensor y soltar encima de la entrada del robot a la que queramos conectar. Por último, se puede cambiar el nombre con el fin de visualizar mejor la función de cada entrada/salida para el momento de programar.

El resultado de la operación anterior se muestra en la siguiente imagen:

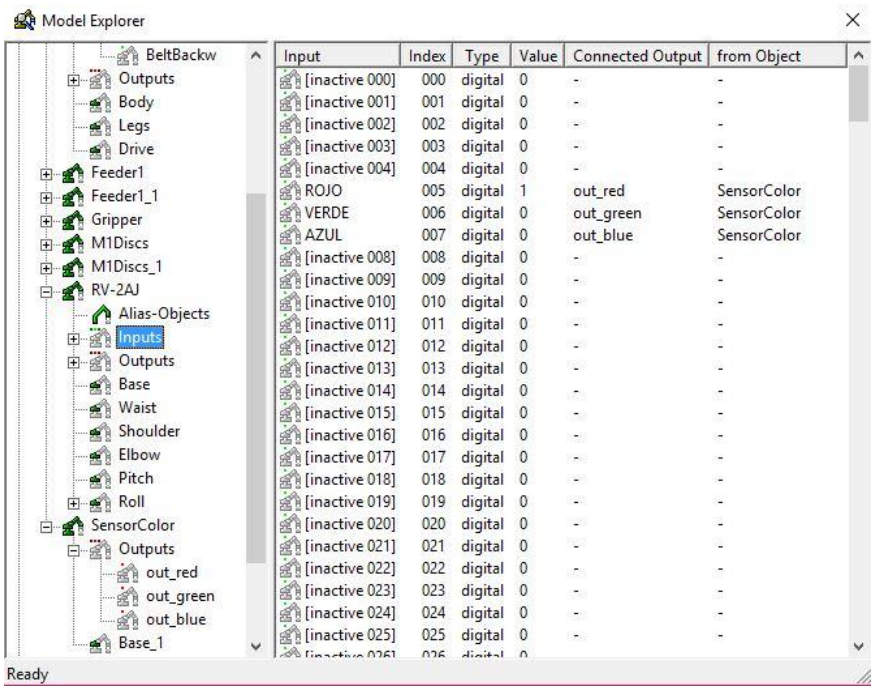
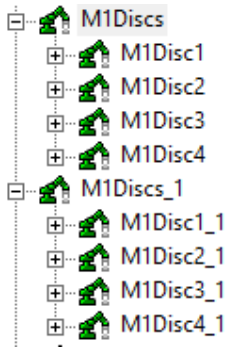


Figura A3.2. Visualización de las entradas del robot (Fuente: Propia).

A continuación, se conectará la señal *BeltO*n, que se encuentra dentro de las entradas del *Conveyor Belt*, con una de las salidas del robot. En este caso se ha escogido la salida 6. Esta señal determinará cuando el robot para el motor que hace funcionar la cinta o, por el contrario, lo pone en marcha.

Se repetirá el proceso con los dos *Feeders*, los creadores de componentes. Se conectará la señal *NextPart*, que se encuentra dentro de las entradas de cada uno de los *Feeders*, con una salida del robot respectivamente. Para este proceso se han escogido las salidas 7 y 8.

Por último, se deben elegir los colores que se quiere que tengan las piezas creadas. Dentro de cada *Feeder*, se encuentran cuatro piezas. Para realizar el cambio de color, se debe abrir el árbol de la opción M1Discs y M1Discs_1 (cada *Feeder* contiene uno).



Clicando con el botón derecho en cada uno de los M1Disc que aparecen, se abrirá un listado de opciones donde se debe seleccionar *Properties*. Dentro de la siguiente ventana, en la opción *Visualization*, se pueden seleccionar los distintos colores.

Nota: Debe elegirse rojo, azul o verde si se quiere que el sensor detecte la pieza, ya que son los únicos colores capaces de ser detectados por él (sensor RGB).

Figura A3.3. Propiedades de cada una de las piezas creadas por el *Feeder*. (Fuente: Propia)

A3.2.3. Grabación de los puntos

Una vez se ha creado el *layout*, se pasa a la creación de los puntos necesarios. En este caso, se deberán grabar 4 puntos en total:

- P1: Punto de recogida de pieza en la cinta.
- P2: Punto de dejada de la pieza roja en la mesa de color rojo.
- P3: Punto de dejada de la pieza verde en la mesa de color verde.
- P4: Punto de dejada de la pieza azul en la mesa de color azul.

El primer punto a crear será el punto de recogida de la pieza en la cinta. Para crear este punto debemos ir a la posición donde se encuentra la caja que hemos colocado como referencia.

Para mover el robot en el espacio y poder llegar a este punto, se debe abrir la ventana de *Teach In*, que se encuentra dentro del menú Extras. Como se explicó en la práctica anterior, se puede mover el robot en los diferentes ejes de coordenadas hasta llegar a la posición deseada.

Una vez se está en la posición donde creamos que el robot agarrará la pieza, se graba el punto clicando sobre el botón *Current Position -> Pos. List*.

A continuación, se cierra la pinza con la opción *Close Hand* que se encuentra en la ventana de *Jog Operatio*n, y se mueve el robot en el espacio como anteriormente hasta grabar los tres puntos de dejada para cada una de las mesas de color.

El resultado de los puntos grabados debe ser el siguiente:

No	Position	Orientation	Comment
P4	-205.0, 325.0, 155.0	-0, 180,R,A	Dejar Azul
P3	-25.0, 325.0, 155.0	0, 180,R,A	Dejar Verde
P2	155.0, 325.0, 155.0	-0, 180,R,A	Dejar Rojo
P1	290.0, 0.0, 145.0	-0, 180,R,A	Coger Pieza

Figura A3.4. Listado de los puntos grabados (Fuente: Propia).

A3.2.4. Programación del proceso

Una vez han sido grabados los puntos necesarios, se debe realizar la programación.

Para ello, es necesario tener claro, primeramente, las instrucciones que se necesitarán por tal de simular el proceso. Como en el caso anterior, estarán presentes las instrucciones MOV, MVS, HOPEN, HCLOSE, DLY, OVRD y SPD, necesarias para realizar los movimientos básicos del robot, abrir y cerrar pinza, así como los cambios de velocidad y tiempos de espera.

Adicionalmente, en este caso, será necesario el uso de condicionales y bucles. Se utilizará el IF para indicar lo que debe hacer el robot si se cumple la condición que indicamos, y en algunos casos qué hacer sino se cumple. Por otro lado, la creación del bucle permitirá que el proceso sea repetido para todas las piezas.

Por otro lado, estarán muy presentes en el programa las entradas y salidas que hemos asignado, por tal de activarlas y desactivarlas según el momento del proceso en el que nos encontremos.

Empezando con la programación, lo primero que se debe hacer es comprobar que el robot tenga la pinza abierta y que se situe en una posición de aproximación al punto de recogida de la pieza. Seguidamente, crear un bucle que se ejecute para todas las piezas, donde los creadores de componentes han de ser activados y desactivados al mismo tiempo, por tal de crear un flag de activación (en nuestro caso, salidas 7 y 8). Seguidamente se debe activar el motor de la cinta, correspondiente a la salida 6.

Una vez las salidas necesarias han sido configuradas, se ha de pasar a los condicionales, definiendo lo siguiente: si uno de los sensores detecta alguna pieza, ya sea roja, verde o azul, la cinta debe ser parada. En el caso de que la condición no se cumpla, el programa debe volver a la línea donde se active el motor de la cinta. Es decir, siempre que no se detecte pieza, la cinta debe seguir en marcha.

En el caso de que sí se haya detectado pieza, será necesario la creación de un *Select/Case*, por tal de definir las instrucciones que deben ser descritas para cada caso dependiendo si la pieza detectada ha sido roja, verde o azul.

En el caso de activarse la entrada que detecta la pieza roja, entrará dentro de las instrucciones definidas en el *Case* correspondiente y así respectivamente para las otras piezas.

El resultado de programa será el siguiente:

```
'--------------------------------------------------------
'            PRÁCTICA 3 - SEÑALES ENTRADA/SALIDA
'    Autor: Sandra Corchado
'    Fecha: Octubre 2017
'--------------------------------------------------------
DEF INTE M1
M1=1
OVRD 100          'Velocidad al 100%
HOPEN 1           'Abrir pinza
MOV P1,-50        'Movimiento hacia un punto de aproximación de P1

WHILE (M1>=1) AND (M1<=8)
M_OUT(7)=1        'Activa Feeder 1
M_OUT(7)=0        'Desactiva Feeder 1
M_OUT(8)=1        'Activa Feeder1_1
M_OUT(8)=0        'Desactiva Feeder1_1
M_OUT(6)=1        'Activa el motor de la cinta
IF M_IN(5)=1 OR M_IN(6)=1 OR M_IN(7)=1 THEN M_OUT(6)=0 ELSE 25


SELECT M_IN(5) AND M_IN(6) AND M_IN(7)

CASE M_IN(5)=1    'Caso en el que se detecte pieza roja
MOV P1, -50       'Movimiento hacia un punto de aproximación de P1
SPD 70            'Velocidad a 70 mm/s
MVS P1            'Movimiento lineal a P1 (coger pieza)
HCLOSE 1          'Cerrar pinza
DLY 1             'Retardo de 1 segundo
MVS P1,-50        'Movimiento lineal hacia un punto de aproximación de P1
SPD 100           'Velocidad a 100 mm/s
MOV P2,-50        'Movimiento hacia un punto de aproximación de P2
SPD 70            'Velocidad a 70 mm/s
MVS P2            'Movimiento lineal a P2 (dejar pieza roja)
HOPEN 1           'Abrir pinza
DLY 1             'Retardo de 1 segundo
MVS P2,-50        'Movimiento lineal hacia un punto de aproximación de P2
SPD 100           'Velocidad a 100 mm/s
MOV P2,-50        'Movimiento hacia un punto de aproximación de P2
MOV P1,-50        'Movimiento hacia un punto de aproximación de P1
M1=M1+1
BREAK


CASE M_IN(6)=1    'Caso en el que se detecte pieza verde
MOV P1, -50       'Movimiento hacia un punto de aproximación de P1
SPD 70            'Velocidad a 70 mm/s
MVS P1            'Movimiento lineal a P1 (coger pieza)
HCLOSE 1          'Cerrar pinza
DLY 1             'Retardo de 1 segundo
MVS P1,-50        'Movimiento lineal hacia un punto de aproximación de P1
SPD 100           'Velocidad a 100 mm/s
MOV P3,-50        'Movimiento hacia un punto de aproximación de P3
SPD 70            'Velocidad a 70 mm/s
MVS P3            'Movimiento lineal a P3 (dejar pieza verde)
HOPEN 1           'Abrir pinza
DLY 1             'Retardo de 1 segundo
MVS P3,-50        'Movimiento lineal hacia un punto de aproximación de P3
SPD 100           'Velocidad a 100 mm/s
MOV P1,-50        'Movimiento hacia un punto de aproximación de P1
M1=M1+1
BREAK
```

```
65 CASE M_IN(7)=1   'Caso en el que se detecte pieza azul
66 MOV P1, -50      'Movimiento hacia un punto de aproximación de P1
67 SPD 70           'Velocidad a 70 mm/s
68 MVS P1           'Movimiento lineal a P1 (coger pieza)
69 HCLOSE 1         'Cerrar pinza
70 DLY 1            'Retardo de 1 segundo
71 MVS P1,-50       'Movimiento lineal hacia un punto de aproximación de P1
72 SPD 100          'Velocidad a 100 mm/s
73 MOV P4,-50       'Movimiento hacia un punto de aproximación de P4
74 SPD 70           'Velocidad a 70 mm/s
75 MVS P4           'Movimiento lineal a P4 (dejar pieza azul)
76 HOPEN 1          'Abrir pinza
77 DLY 1            'Retardo de 1 segundo
78 MVS P4,-50       'Movimiento linal hacia un punto de aproximación de P2
79 SPD 100          'Velocidad a 100 mm/s
80 MOV P1,-50       'Movimiento hacia un punto de aproximación de P1
81 M1=M1+1
82 BREAK
83 END SELECT
84 WEND

85 IF M1>8 THEN 86
86 END
```

A3.2.5. Guardar proyecto, compilación y ejecución

Para guardar el proyecto, se han de repetir los pasos explicados en el punto A1.2.4, correspondientes a la práctica 1. En la opción *Project Management* dentro del menú *Execute*, activar y compilar el archivo .PRJ, es decir, el archivo que contiene la información del proyecto, los archivos .POS y .MBA.

Una vez se ha activado y compilado el proyecto, se puede compilar el programa y comprobar si hay errores. En caso de no encontrar ningún fallo, se puede iniciar la ejecución del programa y visualizar los movimientos en la ventana de animación.

Ampliaciones:

1) Modifica los puntos de dejada para que las piezas no se depositen siempre en la misma posición. Únicamente debe existir un punto y que los demás puntos de dejada sean en función de este.

2) Introduce mesas giratorias en los puntos de dejada, de manera que únicamente sea necesario programar un punto de dejada y activar la mesa para que realice el giro y aparezca de nuevo una posición libre.

Para ello, se debe conectar la entrada *TurnOnce* de la mesa giratoria y conectar con alguna de las salidas disponible del robot.

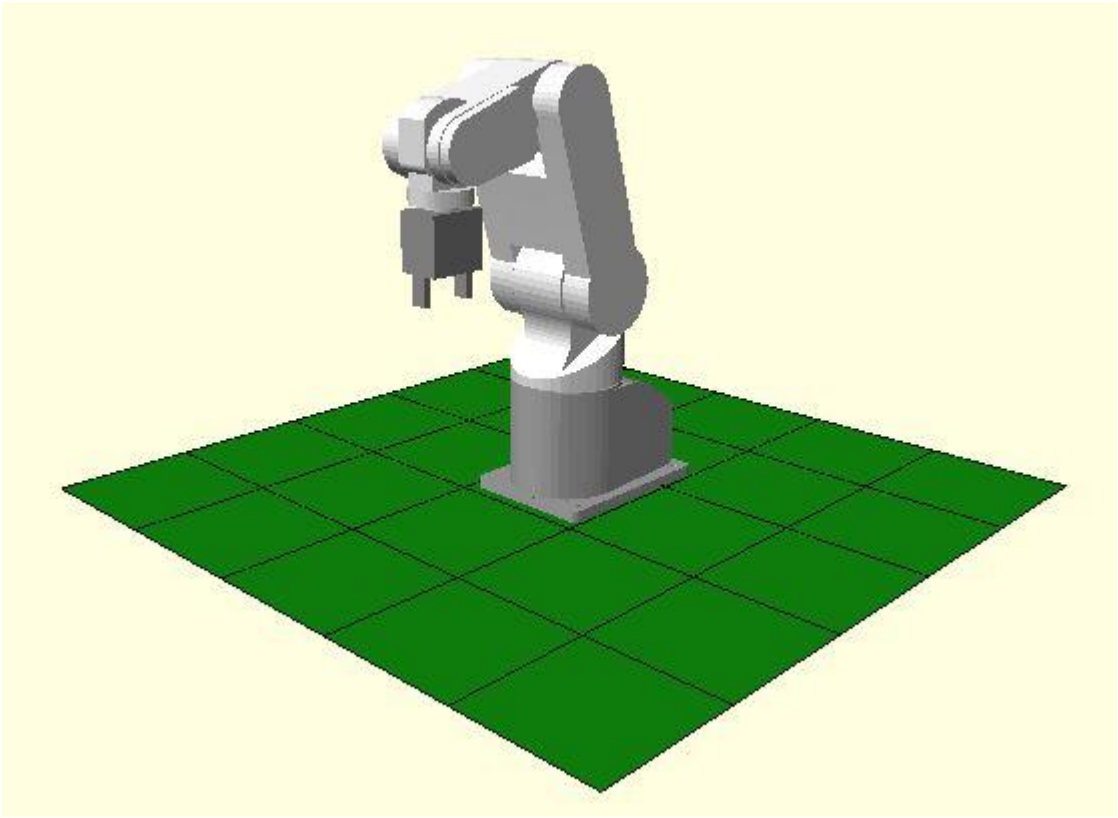
A3.2.6. Cuestiones finales

PRÁCTICA 3: CUESTIONES FINALES	**Fecha:**
Alumno:	**Grupo:**

1. ¿De qué otra manera podría haberse estructurado la programación sin usar la instrucción WHILE? (1 punto)
2. ¿Por qué es necesario activar y desactivar los *Feeders*? (1 punto)
3. Si se analiza la simulación del programa ejecutado, se puede observar que la pieza nunca es dejada en la misma posición en la mesa. ¿Por qué? (1,5 puntos)
4. ¿De qué otra manera propondrías la dejada de las piezas para que sigan siendo clasificadas por colores? (1 punto)
5. ¿Existiría alguna manera de realizar esta programación sin la necesidad de utilizar variables? ¿Cómo sería el programa en ese caso? (1,5 puntos)
6. Si se pidiera disminuir el tiempo ciclo para una aplicación como esta, ¿qué cambios realizarías en esta aplicación? Numéralos explicando uno por uno qué implicarían estos cambios. (2 puntos)
7. Redacta las conclusiones y conceptos aprendidos en la práctica 3. (2 puntos)

A4. PRÁCTICA 4

Automatización y Robótica Industrial



PRÁCTICA 4

Paletizado de cajas

A4.1. Cuestiones previas

PRÁCTICA 4: CUESTIONES PREVIAS	**Fecha:**
Alumno:	**Grupo:**

Para la realización de la práctica 4, es necesario tener claros los conceptos explicados en la práctica 3. También, deben entenderse los conceptos explicados en el punto 11.8 sobre la creación de pallets. Además, también se pueden observar algunos de los vídeos adjuntados en el punto 8.1 de la memoria, donde se ven ejemplos de aplicaciones de paletizado. Esta información se encuentra en el volumen I. Por último, se adjunta un vídeo tutorial mostrando el procedimiento de creación de la práctica:

https://www.youtube.com/watch?v=eB2lcbJDKuY&feature=youtu.be

Una vez observados los vídeos y comprendidos los conceptos explicados, se puede proceder a la resolución de las cuestiones previas:

1. ¿Qué características mecánicas diferencian a los robots paletizadores del resto de modelos de robots? (1 punto)
2. ¿Qué ventaja principal presenta un robot paletizador con respecto al resto de modelos? (1 punto)
3. Dentro del lenguaje de programación Melfa Basic IV, se disponen varios tipos de distribución. Numéralos y explica en qué consiste cada uno de ellos. (1,5 puntos)
4. Dibuja un pallet que venga descrito por la instrucción DEF PLT 1, P2, P3, P4, P5, 4,6,1, indicando la posición de cada uno de los puntos nombrados, y el orden de paletización según la distribución seleccionada. (1,5 puntos)
5. Indica que instrucciones son necesarias por tal de que el robot realice un movimiento a la posición 8 de este pallet. (1 punto)

6. ¿Es posible definir la posición en un pallet en función de una variable? ¿Qué ventaja presenta? (1 punto)
7. ¿Es útil crear una instrucción PLT para un pallet de dimensiones 2x2? (1 punto)
8. ¿Qué es el *flange* de un robot? (1 punto)
9. ¿Qué herramienta suele ser la más adecuada para aplicaciones de paletizado? ¿Por qué? (1 punto)

A4.2. Desarrollo de la práctica

El objetivo de esta práctica es poner en uso las instrucciones de acceso a pallets. Para ello, se desea realizar un programa que recoja las piezas que circulan por una cinta, y según su color las clasifique dentro de un pallet de dimensiones 3x4, de manera que las piezas de color rojo queden posicionadas en la primera fila, las verdes en la fila de en medio, y las azules en la tercera fila. La distribución de las piezas será de tipo 2, para que sigan una misma dirección.

El pallet, que estará situado sobre otra cinta, deberá desaparecer por ella cuando esté completo, es decir, cuando haya paletizado las 12 piezas necesarias, el robot activará dicha cinta.

Como en los casos anteriores, en el acceso a las posiciones de recogida, se deberá contemplar el acercamiento en línea recta desde un punto de aproximación a 50 mm por encima de las mismas a una velocidad de 70 mm/s.

Adicionalmente, se dispondrá de 3 LEDs, uno para cada color a detectar. La finalidad de estos LEDs será encender el correspondiente según el color de la pieza detectada por el sensor de la cinta. El *layout* con todos los componentes será el siguiente:

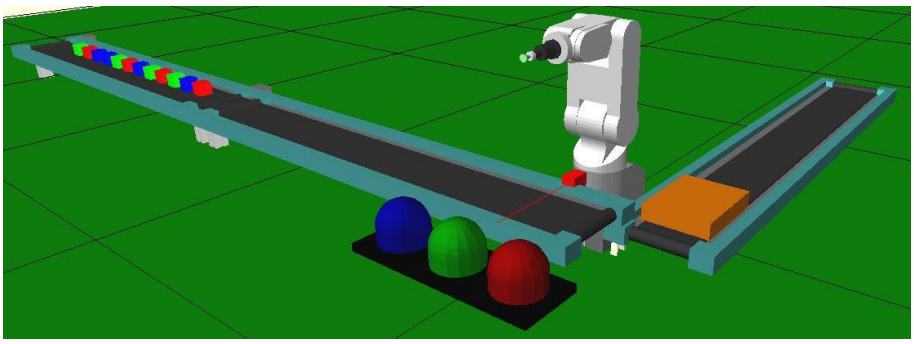


Figura A4.1. Proceso para realizar un paletizado de piezas (Fuente: Propia).

El resultado del proceso se puede observar en la siguiente fotografía:

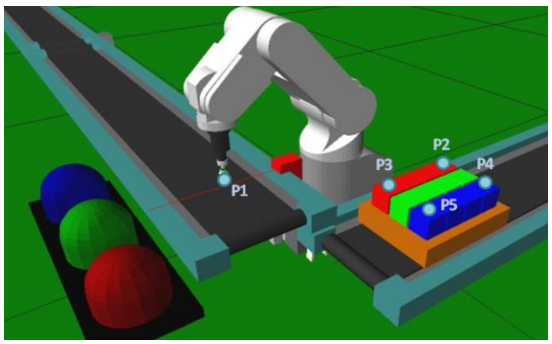


Figura A4.2. Proceso final de realizar un paletizado de piezas (Fuente: Propia).

A4.2.1. Cambio de herramienta

Para realizar un proceso de paletizado, lo más común es utilizar como herramienta una ventosa, ya que permite dejar las piezas de forma más agrupada, mientras que con la pinza no, ya que necesita un espacio para su apertura.

Por tanto, es el primer paso a realizar en la práctica. Para ello, se debe seleccionar la pinza de ventosa, llamada *Vacuum Gripper*, situada en el apartado *Miscellaneous Gripper* que se encuentra en *Model Libraries.*

Una vez insertada, se ha de clicar sobre *Model Explorer On/Off* para poder configurarla. En la ventana que aparece, deben aparecer los elementos presentes en la celda de trabajo del robot, entre los que se encuentra la nueva garra con ventosa.

Para poder adjuntar la nueva garra al plato del robot, se debe desplegar el listado de *Vacuum Gripper* y nuevamente el listado de *Base* que aparece dentro. A continuación, se ha de clicar sobre *Grip Points*, de manera que se obtiene lo siguiente:

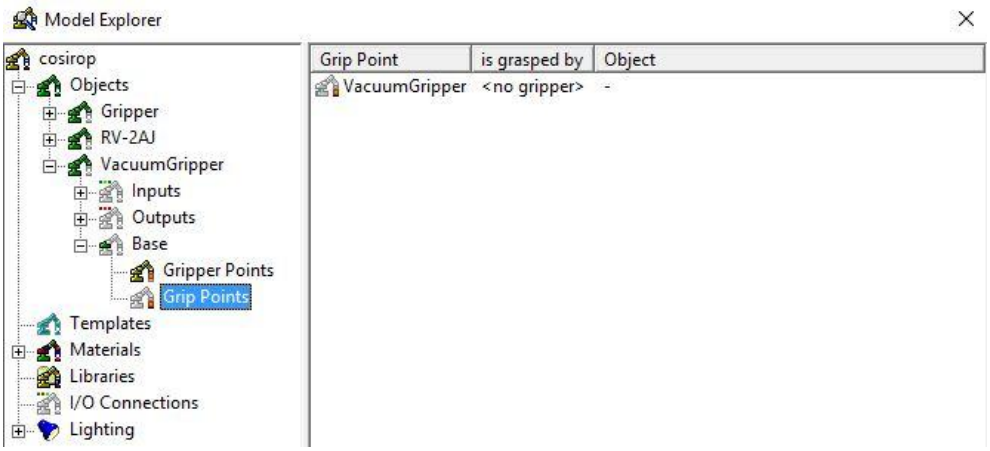


Figura A4.3. Configuración de la garra con ventosa (Fuente: Propia).

Clicando sobre *Vacuum Gripper* con el botón derecho del mouse, y seguidamente seleccionando Propiedades, aparece la siguiente ventana. A continuación, se debe desplegar el listado que aparece en *Gripped by* y seleccionar *Flange*. Automáticamente, la nueva herramienta habrá sido colocada en el plato del robot.

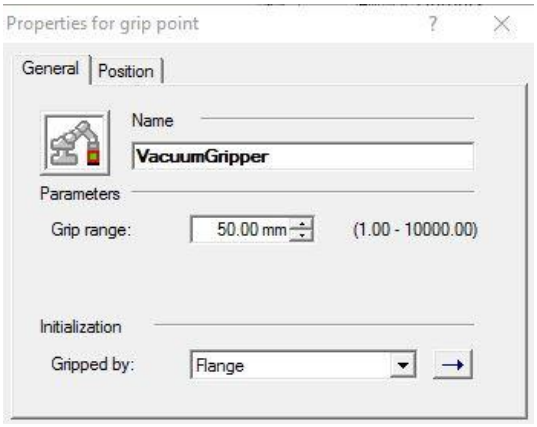


Figura A4.4. Definición de la posición de la garra con ventosa (Fuente: Propia).

A continuación, se realiza una conexión con el robot, por tal de que pueda ser controlada por él. Para ello, se ha de desplegar el listado de *Outputs* del robot, y seleccionar la salida 257, llamada HCLOSE1. Primeramente, se ha de eliminar la conexión existente con la pinza anterior, y para ello, se debe clicar con el botón derecho sobre la salida mencionada y seleccionar *Remove Connection(s)*. A continuación, se pasa a conectar la nueva herramienta, seleccionando la opción *Grasp* que se encuentra desplegando *Inputs*, dentro de *Vacuum Gripper*, y arrastrando esta opción hasta la salida 257 (HCLOSE1).

Una vez realizado este proceso, la nueva garra estará funcionando, pero no podremos observar el funcionamiento hasta que no procedamos a coger una pieza, ya que el movimiento de aire de la ventosa no es posible observarlo en la simulación.

A4.2.2. Creación del *layout*

El listado de las posiciones de los componentes necesarios se presenta en la siguiente tabla:

Tabla A4.1. Posiciones de los componentes del *layout* (Fuente: Propia).

	X (mm)	Y (mm)	Z (mm)
ConveyorBelt	450	-1600	-350
ConveyorBelt_1	450	-3200	-350
ConveyorBelt_2	200	460	-400
Sensor Color	160	0	155

Box ("Pallet")	-150	200	80
LED rojo	450	0	0
LED verde	450	-200	0
LED azul	450	-400	0

Nota: Los primeros dos *conveyors*, permanecerán con la rotación que tienen por defecto al importarlos, pero en el caso del ConveyorBelt_2, se rotará de manera que quede perpendicular a los anteriores. Observar fotografía A4.1.

Por otro lado, las dimensiones para la caja que actúa como pallet son: X 250, Y 200 y Z 50.

Se deben crear las 12 cajas que posteriormente paletizaremos. Todas ellas tendrán un mismo tamaño 50x50x50 mm, y serán colocadas a lo largo del *conveyor* más lejano, con el objetivo final de tener 4 cajas de color rojo, 4 cajas de color azul y 4 cajas de color verde.

Una vez han sido creadas, deben referenciarse dos *Grip Points* a cada una de ellas. Uno de los *Grip Points*, tendrán la función habitual (usado en las anteriores prácticas), por tal de que el robot pueda atrapar las cajas con su *gripper*. Pero en este caso, además de este punto, necesitará uno adicional, por tal de que cuando el pallet marche por la cinta de "salida", las cajas se muevan a su vez con él. Para ello, la cinta nombrada, debe tener una referencia para saber en qué lugar se encuentra cada caja. Por ese motivo, el segundo *Grip Point*, estará situado en la misma posición en X e Y que el primero, pero en Z estará a una altura más pequeña, para que pueda ser reconocido por la segunda cinta, cuando las cajas estén sobre el pallet.

A4.2.3. Grabación de los puntos

Los puntos necesarios para la realización del proceso son los siguientes:

No	Position	Orientation	Comment
P5	55.0, 355.0, 180.0	0, 180,R,A	P5 pallet
P4	-105.0, 355.0, 180.0	-0, 180,R,A	P4 pallet
P3	55.0, 245.0, 180.0	-0, 180,R,A	P3 pallet
P2	-105.0, 245.0, 180.0	0, 180,R,A	P2 pallet
P1	290.0, 0.0, 190.0	0, 180,R,A	Punto inicial

Figura A4.5. Listado de los puntos grabados (Fuente: Propia).

A4.2.4. Programación del proceso

```
10 '------------------------------------------------------------
11 '                PRÁCTICA 4 - PALETIZADO
12 '    Autor: Sandra Corchado
13 '    Fecha: Octubre 2017
14 '------------------------------------------------------------
15 DEF PLT 1,P2,P3,P4,P5,4,3,2
16 DEF INTE M1              'Definición de variables para representar posiciones
17 DEF INTE M2              'del pallet
18 DEF INTE M3
19 DEF INTE M4
20 M1=1
21 M2=5
22 M3=9
23 M4=1
24 '------------------------------------------------------------------

25 OVRD 100
26 HOPEN 1
27 MOV P1, -50
28 M_OUT(6)=1                   'Activación de las cintas de entrada de cajas
29 M_OUT(8)=1

30 WHILE (M4=>1) AND (M4<=12)   'Se ejecutará el bucle para las 12 cajas que
                                'forman el pallet

31 IF M_IN(5)=1 OR M_IN(6)=1 OR M_IN(7)=1 THEN M_OUT(6)=0 AND M_OUT(8)=0 ELSE 28


32 SELECT M_IN(5) AND M_IN(6) AND M_IN(7)

33 CASE M_IN(5)=1           'Caso en el que se detecte pieza roja
34 MOV P1,-50
35 SPD 70
36 MVS P1
37 HCLOSE 1
38 DLY 1
39 MVS P1,-50
40 SPD 100
41 P10=(PLT 1, M1)          'Definición de la posición en el pallet (primera fila)
42 MOV P10, -50             'Movimiento a la posición dentro del pallet
43 SPD 70
44 MVS P10
45 HOPEN 1
46 DLY 1
47 MVS P10,-50
48 SPD 100
49 MOV P1,-50
50 M1=M1+1
51 M4=M4+1
52 BREAK


53 CASE M_IN(6)=1           'Caso en el que se detecte pieza verde
54 MOV P1,-50
55 SPD 70
56 MVS P1
57 HCLOSE 1
58 DLY 1
59 MVS P1,-50
60 SPD 100
61 P11=(PLT 1, M2)          'Definición de la posición en el pallet (segunda fila)
62 MOV P11, -70             'Movimiento a la posición dentro del pallet
63 SPD 70
64 MVS P11
65 HOPEN 1
66 DLY 1
```

```
67 MVS P11,-50
68 SPD 100
69 MOV P1,-50
70 M2=M2+1
71 M4=M4+1
72 BREAK

73 CASE M_IN(7)=1          'Caso en el que se detecte pieza azul
74 MOV P1,-50
75 SPD 70
76 MVS P1
77 HCLOSE 1
78 DLY 1
79 MVS P1,-70
80 SPD 100
81 P12=(PLT 1, M3)         'Definición de la posición en el pallet (tercera fila)
82 MOV P12,-50             'Movimiento a la posición dentro del pallet
83 SPD 70
84 MVS P12
85 HOPEN 1
86 DLY 1
87 MVS P12,-20
88 SPD 100
89 MOV P1,-50
90 M3=M3+1
91 M4=M4+1
92 BREAK
93 END SELECT
94 WEND

95 IF M4>12 THEN M_OUT(7)=1        'Una vez ha sido completado el pallet
                                   'se activa la cinta de salida
96 DLY 15
97 IF (M4>12) THEN 98
98 END
```

A4.2.5. Detección de colisiones

En esta práctica, es de gran importancia controlar si tiene lugar alguna colisión, ya que la dejada de las cajas tiene posiciones muy aproximadas unas respecto a otras y con las fuerzas y velocidades del robot, una pequeña colisión puede dañar el material transportado.

Para realizar la comprobación, se debe activar el detector de colisiones, clicando sobre el icono *Collision Detection* que se encuentra en la barra de herramientas.

Inicialmente se verá que todos los elementos que están en contacto dentro del entorno del robot aparecerán de color rosa. Sin embargo, la importancia de esta comprobación reside en las cajas.

Para ello, debemos iniciar el proceso y asegurar que las cajas no realizan colisiones en el proceso de dejada.

El resultado final sería el siguiente:

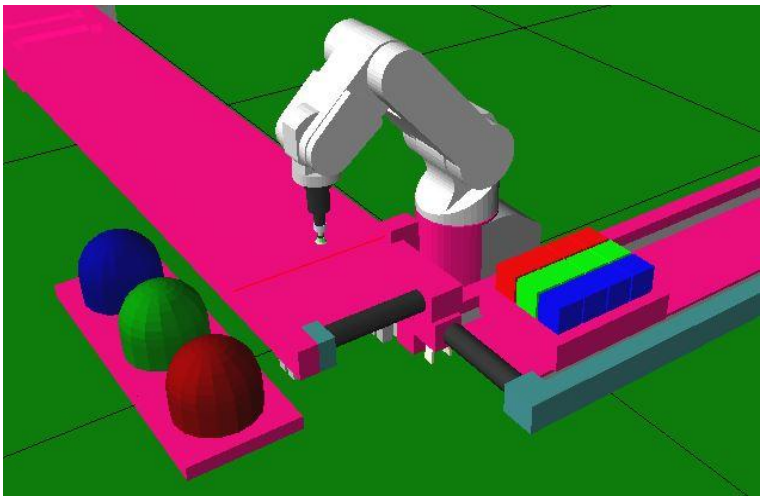


Figura A4.6. Apariencia del *layout* en la comprobación de colisiones (Fuente: Propia).

Ampliaciones:

1) Realiza una segunda capa de piezas, ordenadas por color como anteriormente, sin necesidad de definir una nueva instrucción PLT.

2) Los robots paletizadores, acostumbran a poner un separador entre diferentes capas del pallet. Inserta la geometría y programa los movimientos para que antes de colocar las piezas correspondientes a la segunda capa, se coloque un separador.

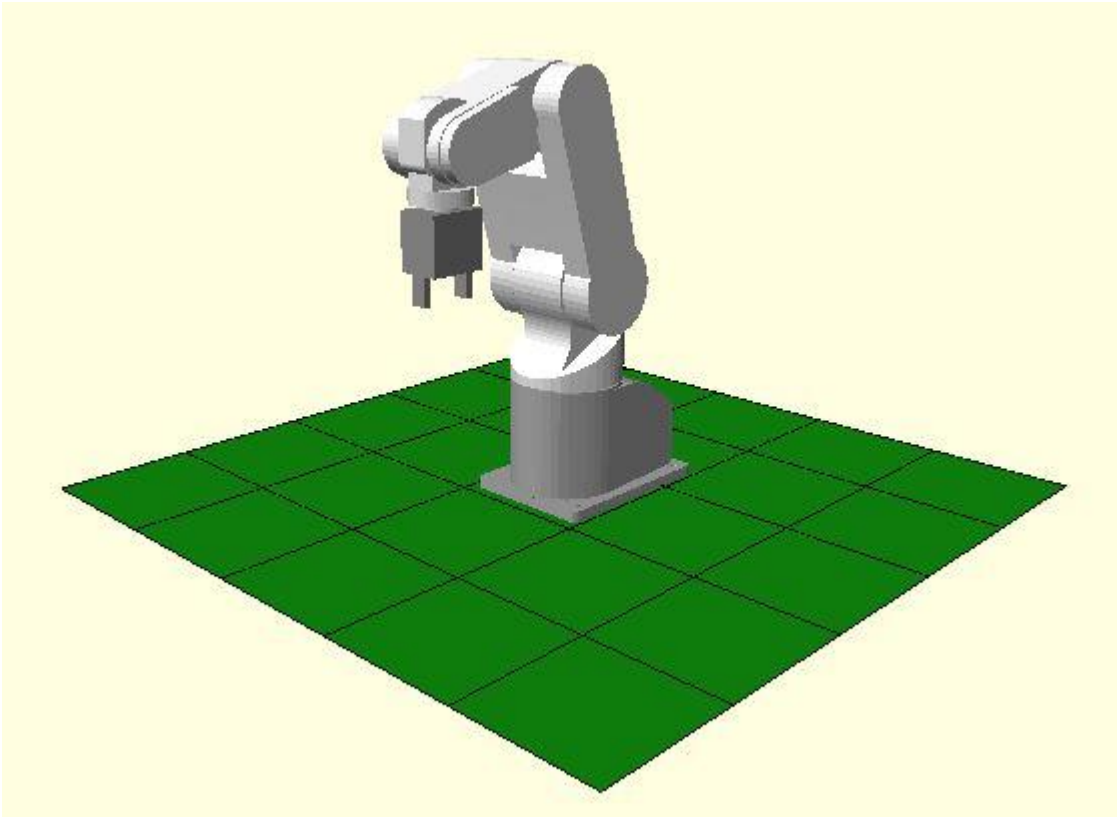
A4.3. Cuestiones finales

PRÁCTICA 4: CUESTIONES FINALES	**Fecha:**
Alumno:	**Grupo:**

1. ¿Qué objetivo tiene la paletización de elementos? (1 punto)
2. En los procesos de paletizado, ¿existe siempre una única solución? Justifica tu respuesta. (1,5 puntos)
3. ¿De qué manera se puede optimizar tiempos de ciclo en el paletizado? (1,5 puntos)
4. ¿Qué estrategias se pueden tomar a la hora de paletizar bloques uniformes? (1 punto)
5. En las aplicaciones industriales, ¿cómo suele realizarse la entrada de los pallets a la posición de dejada? (1 punto)
6. ¿En qué posiciones tiene mayor importancia la comprobación de colisiones en la paletización? (1 punto)
7. Numera las ventajas que introduce la paletización en el mundo industrial. (1 punto)
8. Redacta las conclusiones y conceptos aprendidos en la práctica 4. (2 puntos)

A5. PRÁCTICA 5

Automatización y Robótica Industrial



PRÁCTICA 5

Subrutinas

A5.1. Cuestiones previas

PRÁCTICA 5: CUESTIONES PREVIAS	**Fecha:**
Alumno:	**Grupo:**

Para la realización de la práctica 5 deben entenderse los conceptos explicados en el punto 11.9 sobre el objetivo de la creación y llamadas a subrutina, disponibles en el volumen I. Por otro lado, se adjunta un vídeo tutorial mostrando el procedimiento de creación de la práctica:

https://www.youtube.com/watch?v=Yd1HsIxGgsY

Una vez observado el vídeo tutorial y comprendidos los conceptos explicados, se puede proceder a la resolución de las cuestiones previas:

1. ¿Cuál es el objetivo principal de crear una subrutina? (1 punto)
2. ¿Con qué instrucción debe ser finalizada la subrutina para que el programa pueda volver a la posición donde estaba previamente? (1 punto)
3. ¿Una misma subrutina puede llamar a otra? ¿Y que esta otra introduzca otra nueva llamada a otra subrutina diferente? En el caso de que sea posible, pon un ejemplo. (1,5 puntos)
4. Dentro de la estructura de la programación, ¿dónde se escriben las subrutinas? (1 punto)
5. ¿Podría existir una subrutina denominada "Abrir pinza" y "Cerrar pinza"? ¿Por qué? (1 punto)
6. Para la distinción de 4 piezas, una roja, azul, verde y amarillo, ¿qué sensor o sensores son necesarios? ¿Cómo programarías la instrucción principal para distinguir cada una de las piezas? (1,5 puntos)

7. Si existen 4 piezas de diferentes alturas, ¿es necesario programar 4 puntos de recogida o existe alguna otra posibilidad? (1,5 puntos)
8. Pon un ejemplo de alguna aplicación en la que sea útil el uso de subrutinas. (1,5 puntos)

A5.2. Desarrollo de la práctica

El objetivo de esta práctica es poner en uso la creación y posteriormente la llamada de subrutinas. Para ello, se plantean unas tareas repetitivas de manera que deban programarse subrutinas por tal de evitar repetir código.

En esta práctica se incluirán dos procesos a realizar por dos robots respectivamente. El primero de ellos consistirá en poner tapones a unas botellas que van llegando a través de una cinta. Como en los casos anteriores, un sensor detectará la botella y el robot hará parar la cinta mediante el uso de señales. Según el color de la botella, la altura variará, siguiendo de mayor a menor el orden siguiente: Roja > Amarilla > Verde > Azul. Una vez detectada la botella, el robot irá a buscar su correspondiente tapón, para posteriormente colocarlo sobre la botella. En este proceso, se encuentra un proceso común para todas las botellas, el momento en el que el robot empuja el tapón para poder ser colocado. Por tanto, se creará una subrutina, que será llamada para todas las botellas, denominada *"Empujar_tapón".*

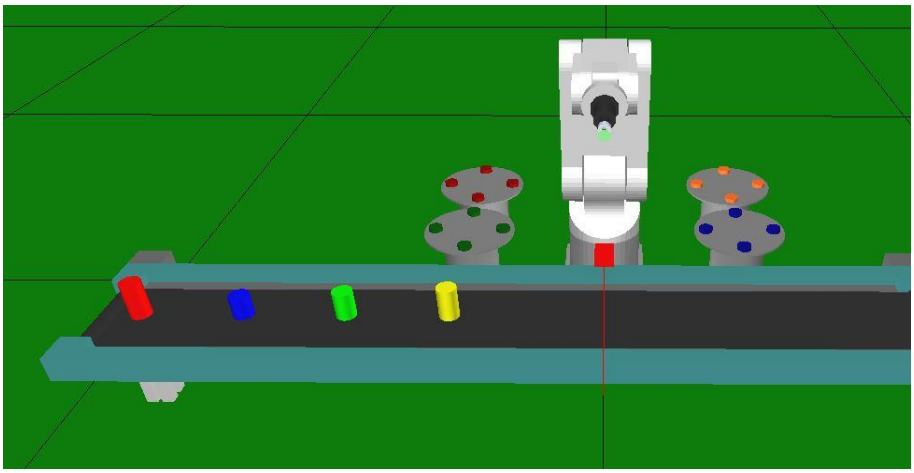


Figura A5.1. Primer robot en el proceso de poner tapones (Fuente: Propia).

Una vez los tapones han sido colocados, las botellas avanzarán por una cinta hasta llegar al final de esta, donde se encontrará un robot encargado de clasificar cada una de las botellas en la cinta correspondiente. En este caso, también se tiene un proceso común para todas las botellas, que consiste en la recogida de cada una. Por tanto, para este robot también se dispondrá de una subrutina denominada *"Coger_Botella".*

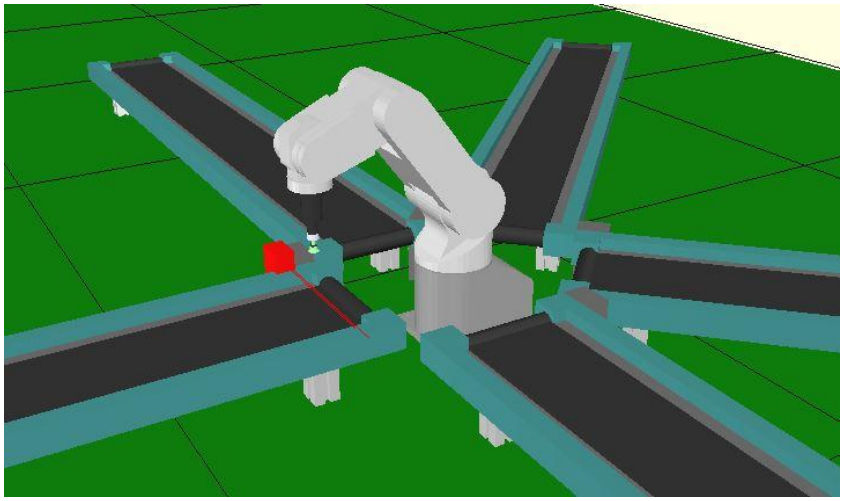


Figura A5.2. Segundo robot en el proceso de clasificación de las botellas (Fuente: Propia).

Como en los casos anteriores, en el acceso a las posiciones de recogida, deberá contemplarse el acercamiento en línea recta desde un punto de aproximación a 50 mm por encima de las mismas a una velocidad de 70 mm/s.

A5.2.1. Creación del *layout*

El listado de las posiciones de los componentes necesarios se presenta en la siguiente tabla:

Tabla A5.1. Posiciones de los componentes del *layout* (Fuente: Propia).

	Posición			Rotación		
	X (mm)	**Y (mm)**	**Z (mm)**	**Roll (°)**	**Pitch (°)**	**Yaw (°)**
RV-2AJ	0	0	0	0	0	0
RV-2AJ_1	290	2600	0	-90	0	0
ConveyorBelt	450	-1000	-350	90	0	0
ConveyorBelt_1	450	600	-350	90	0	0
ConveyorBelt_2	500	2400	-350	0	0	0
ConveyorBelt_3	100	2700	-350	180	0	0
ConveyorBelt_4	500	2700	-350	45	0	0

ConveyorBelt_5	300	2900	-350	135	0	0
TurnTable	0	300	0	0	0	0
TurnTable_1	0	-300	0	0	0	0
TurnTable_2	-220	300	0	0	0	0
TurnTable_3	-220	-300	0	0	0	0
SensorColor	140	0	150	0	90	0
SensorColor_1	150	2300	165	0	90	0
SensorDistance	140	0	150	0	90	0
SensorDistance_1	150	2300	170	0	90	0

Por otro lado, se dispone de los cilindros que simularán las botellas y los tapones de las botellas. A continuación, se muestran las medidas en la siguiente tabla:

Tabla A5.2. Medidas de los cilindros del *layout* (Fuente: Propia).

	Radio (mm)	**Altura (mm)**
Cylinder (Botella roja)	20	80
Cylinder_1 (Botella amarilla)	20	70
Cylinder_2 (Botella verde)	20	60
Cylinder_3 (Botella azul)	20	50
16x Cylinder (Tapones)	15	10

Las posiciones de los cilindros que representan las botellas quedarán colocados previamente a la llegada del sensor, mientras que los tapones quedarán sobre sus correspondientes tablas giratorias, tal y como se puede observar en la fotografía A5.1.

A5.2.2. Grabación de los puntos

Los puntos necesarios para la realización del proceso para el caso del primer robot (RV-2AJ), son los siguientes:

No	Position	Orientation	Comment
P5	-145.0,-300.0, 165.0	117, 180,R,A	Coger tapón rojo
P4	75.0,-300.0, 165.0	117, 180,R,A	Coger tapón verde
P3	-145.0, 300.0, 165.0	-90, 180,R,A	Coger tapón naranja
P2	75.0, 300.0, 165.0	-90, 180,R,A	Coger tapón azul
P1	290.0, -10.0, 250.0	0, 180,R,A	Punto inicial/Poner tapón

Figura A5.3. Listado de los puntos grabados para el robot RV-2AJ (Fuente: Propia).

Por otro lado, los puntos necesarios para la segunda parte del proceso, realizado por el segundo robot (RV-2AJ_2) son los siguientes:

No	Position	Orientation	Comment
P5	-280.0,-170.0, 210.0	117, 180,R,A	Dejar botella roja
P4	40.0,-300.0, 190.0	117, 180,R,A	Dejar botella verde
P3	-280.0, 170.0, 200.0	-90, 180,R,A	Dejar botella amarilla
P2	40.0, 315.0, 180.0	-90, 180,R,A	Dejar botella azul
P1	300.0, 0.0, 250.0	0, 180,R,A	Punto inicial/Coger botella

Figura A5.4. Listado de los puntos grabados para el robot RV-2AJ_2 (Fuente: Propia).

A.5.2.3. Programación del proceso

Para realizar la programación de este proceso, será necesario el uso de la instrucción *Select/Case,* explicada en el punto 11.4.2 de la memoria disponible en el volumen I, que permite indicar al robot la acción que debe hacer según la condición que se ha dado. Es adecuado para este caso ya que tenemos 4 tipos de condiciones.

Por otro lado, cabe recordar lo que se ha comentado en el enunciado, el uso necesario de las subrutinas "*Empujar_Tapón"* y "*Coger_Botella"*, por tal de ahorrar código de programa.

De esta manera, el resultado de la programación para el primer robot (RV-2AJ) es el siguiente:

```
'-------------------------------------------------------------------------------------------
'                            PRÁCTICA 5 - SUBRUTINAS
'    Autor: Sandra Corchado
'    Fecha: Noviembre 2017
'-------------------------------------------------------------------------------------------

DEF INTE M1              'Creación de la variable entera M1
M1=1                     'Inicialización de M1

HOPEN 1
OVRD 100
MOV P1, -50
M_OUT(6)=1               'Activar cinta

WHILE (M1>=1 AND M1<=4)          'Mientras que M1 esté entre 1 y 4 entra en el bucle
IF M_IN(8)=1 THEN M_OUT(6)=0 ELSE GOTO 20        'Si se activa el sensor de presencia, se apaga la cinta,
                                                 'sino se mantiene encendida (salto linea 15)
SELECT M_IN(5) AND M_IN(6) AND M_IN(7) AND M_IN(8)
CASE M_IN(5)= 1                  'Si se activa la señal 5 (detecta botella roja) realiza las siguientes acciones
MOV P5, -50
SPD 70
MVS P5
HCLOSE 1
DLY 1
MVS P5, -50
SPD 100
MOV P1,-50
M1=M1+1
P1.Z=210                         'Establece posición en Z para dejada del tapón
GOSUB *EMPUJAR_TAPON             'Llamada a subrutina
M_OUT(6)=1
DLY 1
BREAK


CASE M_IN(6)= 1                  'Si se activa la señal 6 (detecta botella verde) realiza las siguientes acciones
MOV P4, -50
SPD 70
MVS P4
HCLOSE 1
DLY 1
MVS P4, -50
SPD 100
MOV P1,-50
M1=M1+1
P1.Z=190                         'Establece posición en Z para dejada del tapón
GOSUB *EMPUJAR_TAPON             'Llamada a subrutina
M_OUT(6)=1
DLY 1
BREAK

CASE M_IN(7)=1                   'Si se activa la señal 7 (detecta botella azul) realiza las siguientes acciones
MOV P2, -50
SPD 70
MVS P2
HCLOSE 1
DLY 1
MVS P2, -50
SPD 100
MOV P1, -50
M1=M1+1
P1.Z=180                         'Establece posición en Z para dejada del tapón
GOSUB *EMPUJAR_TAPON             'Llamada a subrutina
M_OUT(6)=1
DLY 1
BREAK


CASE M_IN(8)=1                  'Si se activa la señal 8 (detecta botella amarilla) realiza las siguientes acciones
MOV P3, -50
SPD 70
MVS P3
HCLOSE 1
DLY 1
MVS P3, -50
SPD 100
MOV P1,-50
M1=M1+1
P1.Z=200                        'Establece posición en Z para dejada del tapón
GOSUB *EMPUJAR_TAPON            'Llamada a subrutina
M_OUT(6)=1
DLY 1
BREAK
END SELECT
WEND
```

```
87 IF M1>4 THEN GOTO 88             'Si la variable M1 supera 4, entonces salto a la linea 88 y fin del programa
88 END

90 *EMPUJAR_TAPON                   'Subrutina Empujar_Tapon
91 MVS P1
92 HOPEN 1
93 DLY 1
94 MVS P1,-50
95 RETURN
```

Por otro lado, el resultado de la programación para el primer robot (RV-2AJ_2) es el siguiente:

```
10 '-----------------------------------------------------------------------------------------
11 '                            PRÁCTICA 5 - SUBRUTINAS
12 '    Autor: Sandra Corchado
13 '    Fecha: Noviembre 2017
14 '-----------------------------------------------------------------------------------------

15 DEF INTE M1              'Creación de la variable entera M1
16 M1=1                     'Inicialización de M1

17 HOPEN 1
18 OVRD 100
19 MOV P1, -50
20 M_OUT(7)=1               'Activa cinta de salida
21 M_OUT(8)=1               'Activa cinta de salida
22 M_OUT(9)=1               'Activa cinta de salida
23 M_OUT(10)=1              'Activa cinta de salida
24 M_OUT(6)=1               'Activa cinta de entrada de material

25 WHILE (M1>=1 AND M1<=4)              'Mientras que M1 esté entre 1 y 4 entra en el bucle
26 IF M_IN(8)=1 THEN M_OUT(6)=0 ELSE GOTO 24        'Si se activa el sensor de presencia, se apaga la cinta,
27                                                  'sino se mantiene encendida (salto linea 24)
28 SELECT M_IN(5) AND M_IN(6) AND M_IN(7) AND M_IN(8)
29 CASE M_IN(5)= 1                  'Si se activa la señal 5 (detecta botella roja) realiza las siguientes acciones
30 P1.Z=210                         'Establece posición en Z para dejada del tapón
31 GOSUB *COGER_BOTELLA             'Llamada a subrutina
32 MOV P5, -50
33 SPD 70
34 MVS P5
35 DLY 1
36 HOPEN 1
37 MVS P5, -50
38 SPD 100
39 MOV P1,-50
40 M1=M1+1
41 GOTO 24
42 BREAK


43 CASE M_IN(6)= 1                  'Si se activa la señal 6 (detecta botella verde) realiza las siguientes acciones
44 P1.Z=190                         'Establece posición en Z para dejada del tapón
45 GOSUB *COGER_BOTELLA             'Llamada a subrutina
46 MOV P4, -50
47 SPD 70
48 MVS P4
49 DLY 1
50 HOPEN 1
51 MVS P4, -50
52 SPD 100
53 MOV P1,-50
54 M1=M1+1
55 GOTO 24
56 BREAK

57 CASE M_IN(7)=1                   'Si se activa la señal 7 (detecta botella azul) realiza las siguientes acciones
58 P1.Z=180                         'Establece posición en Z para dejada del tapón
59 GOSUB *COGER_BOTELLA             'Llamada a subrutina
60 MOV P2, -50
61 SPD 70
62 MVS P2
63 DLY 1
64 HOPEN 1
65 MVS P2, -50
66 SPD 100
67 MOV P1, -50
68 M1=M1+1
69 GOTO 24
70 BREAK
```

```
43 CASE M_IN(6)= 1                'Si se activa la señal 6 (detecta botella verde) realiza las siguientes acciones
44 P1.Z=190                       'Establece posición en Z para dejada del tapón
45 GOSUB *COGER_BOTELLA           'Llamada a subrutina
46 MOV P4, -50
47 SPD 70
48 MVS P4
49 DLY 1
50 HOPEN 1
51 MVS P4, -50
52 SPD 100
53 MOV P1,-50
54 M1=M1+1
55 GOTO 24
56 BREAK

57 CASE M_IN(7)=1                 'Si se activa la señal 7 (detecta botella azul) realiza las siguientes acciones
58 P1.Z=180                       'Establece posición en Z para dejada del tapón
59 GOSUB *COGER_BOTELLA           'Llamada a subrutina
60 MOV P2, -50
61 SPD 70
62 MVS P2
63 DLY 1
64 HOPEN 1
65 MVS P2, -50
66 SPD 100
67 MOV P1, -50
68 M1=M1+1
69 GOTO 24
70 BREAK


71 CASE M_IN(8)=1                 'Si se activa la señal 8 (detecta botella amarilla) realiza las siguientes acciones
72 P1.Z=200                       'Establece posición en Z para dejada del tapón
73 GOSUB *COGER_BOTELLA           'Llamada a subrutina
74 MOV P3, -10
75 SPD 70
76 MVS P3
77 DLY 1
78 HOPEN 1
79 MVS P3, -50
80 SPD 100
81 MOV P1,-50
82 M1=M1+1
83 GOTO 24
84 BREAK
85 END SELECT
86 WEND

87 IF M1>4 THEN GOTO 88           'Si la variable M1 supera 4, entonces salto a la linea 88 y fin del programa
88 END

89 *COGER_BOTELLA                 'Subrutina Coger_Botella
90 MOV P1, -50
91 SPD 70
92 MVS P1
93 HCLOSE 1
94 DLY 1
95 MVS P1, -50
96 SPD 100
97 RETURN
```

Ampliaciones:

1) Conectar las tablas giratorias para que una vez se haya recogido un tapón de la mesa, la tabla gire y un nuevo tapón sea colocado en la posición de recogida del robot.

2) Introduce un nuevo robot, seguido del que pone los tapones, que realice una nueva subrutina, como por ejemplo poner una pegatina a cada botella según el color y tamaño de cada una.

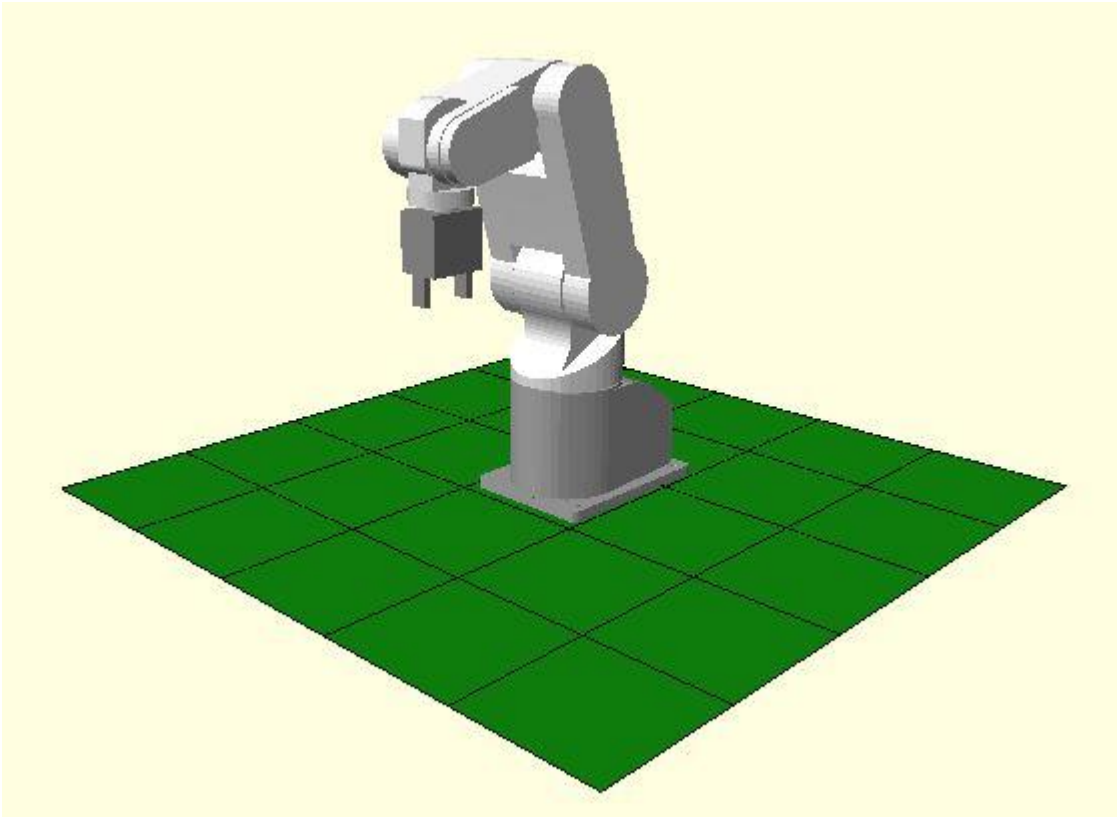
A5.3. Cuestiones finales

PRÁCTICA 5: CUESTIONES FINALES	**Fecha:**
Alumno:	**Grupo:**

1. Realice un informe donde se explique la función del nuevo robot, la programación realizada y un vídeo donde se observe el proceso final.

A6. PRÁCTICA 6

Automatización y Robótica Industrial



PRÁCTICA 6

Interrupciones

A6.1. Cuestiones previas

PRÁCTICA 6: CUESTIONES PREVIAS	**Fecha:**
Alumno:	**Grupo:**

Para la realización de las cuestiones previas, es necesario entender los conceptos explicados en el punto 11.10 de la memoria (volumen I). Por otro lado, se adjunta un vídeo tutorial donde se representa la creación de una práctica con el uso de interrupciones:

https://www.youtube.com/watch?v=w3uiVBxHHyM&feature=youtu.be

Una vez se han asumido los conceptos explicados y se ha observado el vídeo, se puede proceder a la realización de las cuestiones previas:

1. ¿Cuál es el objetivo principal de crear una interrupción? (1 punto)
2. ¿Cómo se define una interrupción en el lenguaje Melfa Basic IV? (1 punto)
3. ¿Qué tipos de condiciones pueden activar una interrupción? (1 punto)
4. Si se dan dos interrupciones al mismo tiempo, ¿cuál de ellas se activa primero? Indica cómo definir la prioridad en interrupciones. (1 punto)
5. ¿Es posible deshabilitar una interrupción en un punto determinado del programa? Si es posible, indica cómo hacerlo. (1 punto)
6. Numera los diferentes tipos de volver al programa principal después de producirse una interrupción y las diferencias entre ellos. (1 punto)

7. Si un robot está realizando agujeros y se activa una señal externa para que el robot realice una interrupción para realizar otro agujero en una nueva pieza situada en otro lugar, ¿sería posible realizar la interrupción? (1 punto)
8. ¿Cuántas interrupciones pueden ser definidas en un mismo programa? (1 punto)
9. ¿Qué variable existente en el lenguaje Melfa Basic IV permite contabilizar tiempo real? (1 punto)
10. Pon un ejemplo de una aplicación en la que sea viable el uso de interrupciones. (1 punto)

A6.2. Desarrollo de la práctica

El objetivo de esta práctica es poner un ejemplo de cómo sería el uso de las instrucciones de interrupción. Un ejemplo de una aplicación en la que se podría dar el uso de interrupciones es la siguiente:

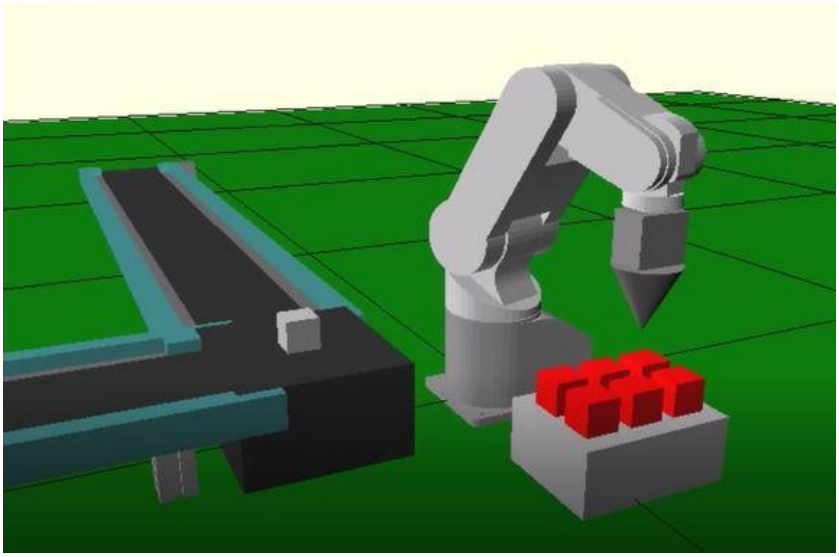


Figura A6.1. Ejemplo de *layout* en un proyecto con la presencia de interrupciones (Fuente: Propia).

Se tiene un robot aplicando cola a las piezas que aparecen sobre la mesa. Mientras el robot está realizando estas acciones, por la cinta de su lado aparecerán materiales a través de una cinta hasta llegar a la caja de color negro. Suponiendo que la caja de color negro sea una máquina que debe realizar algún proceso sobre el cubo gris, se necesitará un tiempo determinado hasta que la pieza esté lista. Una vez haya pasado este tiempo, se activará una interrupción, en la que el robot dejará de aplicar la cola a las cajas rojas, y pasará a aplicar cola a la caja gris.

Una vez el robot finalice la aplicación de poner cola sobre la caja gris, volverá al trabajo que se estaba realizando previamente a la llamada de interrupción, hasta nuevo aviso.

La finalidad del uso de interrupciones es evitar tiempos de espera innecesarios, haciendo que el robot realice otras funciones mientras que otros procesos se completan sin la ayuda de este.

A6.2.1. Creación del *layout*

Las dimensiones y posiciones de las cajas situadas en el *layout* se representan en la siguiente tabla:

Tabla A6.1. Posiciones de los componentes del *layout* (Fuente: Propia).

	Dimensiones			Posición		
	X (mm)	**Y (mm)**	**Z (mm)**	**X (mm)**	**Y (mm)**	**Z (mm)**
Box (mesa)	230	180	100	-150	200	0
Box_1 (Caja roja)	50	50	50	-130	300	100
Box_2 (Caja roja)	50	50	50	-60	300	100
Box_3 (Caja roja)	50	50	50	10	300	100
Box_4 (Caja roja)	50	50	50	10	230	100
Box_5 (Caja roja)	50	50	50	-60	230	100
Box_6 (Caja roja)	50	50	50	-130	230	100
Box_7 (Máquina)	250	280	150	200	-150	0
Box_8 (Caja gris)	50	50	50	300	-30	150

Por otro lado, las posiciones de las cintas de entrada y salida del material son las siguientes:

Tabla A6.3. Posiciones de los componentes del *layout* (Fuente: Propia).

	Posición			Rotación		
	X (mm)	**Y (mm)**	**Z (mm)**	**Roll (°)**	**Pitch (°)**	**Yaw (°)**
ConveyorBelt	2190	150	-350	180	0	0
ConveyorBelt_1	500	-1900	-350	90	0	0

A6.2.2. Grabación de los puntos

Para este caso se han grabado los puntos necesarios para completar el recorrido del contorno de una caja roja, y posteriormente en la programación se ha añadido un offset a estos puntos para poner realizar el mismo proceso para todas las cajas y de esta manera disminuir la cantidad de puntos a crear. Por otro lado, se han creado también los puntos necesarios para realizar el recorrido por la caja gris, de manera que el resultado total de puntos creados es el siguiente:

No	Position	Orientation	Comment
P9	350.0, 20.0, 250.0	0, 180,R,A	Vértice 4 cuadrado 2
P8	350.0, -30.0, 250.0	0, 180,R,A	Vértice 3 cuadrado 2
P7	300.0, -30.0, 250.0	0, 180,R,A	Vértice 2 cuadrado 2
P6	300.0, 20.0, 250.0	0, 180,R,A	Vértice 1 cuadrado 2
P5	-80.0, 230.0, 200.0	-90, 180,R,A	Vértice 4 cuadrado
P4	-80.0, 280.0, 200.0	-90, 180,R,A	Vértice 3 cuadrado
P3	-130.0, 280.0, 200.0	-90, 180,R,A	Vértice 2 cuadrado
P2	-130.0, 230.0, 200.0	-90, 180,R,A	Vértice 1 cuadrado
P1	0.0, 290.0, 250.0	-90, 180,R,A	Punto inicial

Figura A6.2. Listado de los puntos grabados (Fuente: Propia).

A6.2.3. Programación del proceso

Para este tipo de programación, es necesario el uso de entradas externas o contadores. Puede ser que un pulsador envíe una señal al robot de que el proceso sobre el cubo rojo ha finalizado y entonces activar la interrupción. Pero normalmente, se tiene que un contador (*timer*), contabiliza el tiempo transcurrido desde que la pieza ha llegado a la estación de procesado, y una vez ha pasado un tiempo determinado se activa la señal de interrupción.

Para ello, es necesario disponer en el programa de entradas externas o alguna instrucción que realice la función de contador. En este caso, se dispone de una variable denominada M_TIMER(x) que realiza un contador de tiempo real.

Utilizando este contador, se puede definir la interrupción de manera que cuando haya pasado un tiempo determinado, por ejemplo 10 segundos, se produzca la interrupción y el robot cambie la actividad que estaba realizando. Cabe destacar que esta variable contabiliza el tiempo en milisegundos. De esta manera, el resultado de la programación es el siguiente:

```
'------------------------------------------------------------------------------
'                  PRÁCTICA 6 - INTERRUPCIONES
'   Autor: Sandra Corchado
'   Fecha: Noviembre 2017
'------------------------------------------------------------------------------
10 DEF POS OFFSETX              'Definición de un offset en X
11 DEF POS OFFSETY              'Definición de un offset en Y
12 OFFSETX=(+70,0,0,0,0,0)
13 OFFSETY=(0,+70,0,0,0,0)
14 DEF ACT 1, M_TIMER(1)>10000 GOSUB *PEGAMENTO_PIEZA2      'Definición de la interrupción

15 M_TIMER(1)=0                 'Inicialización de la variable Timer
16 ACT 1 = 0                    'Deshabilita la interrupción

17 HCLOSE 1
18 OVRD 100

19 ACT 1 = 1                    'Habilita la interrupción

20 WHILE 1
21 MOV P1
22 GOSUB *PONER_PEGAMENTO       'Llamada a subrutina Poner_Pegamento
23 P2=P2+OFFSETX                'Variación de las posiciones añadiendo Offset
24 P3=P3+OFFSETX
25 P4=P4+OFFSETX
26 P5=P5+OFFSETX
27 GOSUB *PONER_PEGAMENTO
28 P2=P2+OFFSETX
29 P3=P3+OFFSETX
30 P4=P4+OFFSETX
31 P5=P5+OFFSETX
32 GOSUB *PONER_PEGAMENTO
33 P2=P2+OFFSETY
34 P3=P3+OFFSETY
35 P4=P4+OFFSETY
36 P5=P5+OFFSETY


37 GOSUB *PONER_PEGAMENTO
38 P2=P2-OFFSETX
39 P3=P3-OFFSETX
40 P4=P4-OFFSETX
41 P5=P5-OFFSETX
42 GOSUB *PONER_PEGAMENTO
43 P2=P2-OFFSETX
44 P3=P3-OFFSETX
45 P4=P4-OFFSETX
46 P5=P5-OFFSETX
47 GOSUB *PONER_PEGAMENTO
48 P2=P2-OFFSETY
49 P3=P3-OFFSETY
50 P4=P4-OFFSETY
51 P5=P5-OFFSETY
52 WEND

53 *PONER_PEGAMENTO             'Subrutina poner pegamento a las cajas rojas
54 MOV P2, -50
55 SPD 70
56 MVS P2
57 MVS P3
58 MVS P4
59 MVS P5
60 MVS P2
61 MVS P2, -50
62 SPD 100
63 RETURN
```

```
64 *PEGAMENTO_PIEZA2            'Subrutina poner pegamento a la caja gris (interrupción)
65 M_TIMER(1)=0
66 ACT 1 = 0
67 MOV P6,-50
68 MVS P6
69 MVS P7
70 MVS P8
71 MVS P9
72 MVS P6
73 MVS P6, -50
74 ACT 1 = 1
75 RETURN
```

Para que la simulación se adaptara a un proceso real de interrupción, el robot, una vez ejecutadas las instrucciones indicadas en la subrutina de interrupción, debería volver al lugar donde se produjo la interrupción, ya que, no es correcto que el robot deje una de las aristas de la caja sin aplicar la cola. Para ello, la interrupción debería ser finalizada con la instrucción RETURN 0 que según la teoría indicada en el apartado 11.10.1 de la memoria (volumen I) y comprobado por la ayuda de Cosimir®, permite volver al lugar donde se produce la interrupción. Sin embargo, si se escribe el 0 a continuación del RETURN, Cosimir® no es capaz de compilar el programa, de manera que los datos no se cargan al robot y, por tanto, cuando el robot termina de ejecutar los movimientos correspondientes al proceso de interrupción, vuelve a la línea siguiente del proceso.

Ampliaciones:

1) Modifica el programa de manera que existan dos robots, y que sea el segundo robot el que envíe una señal para que se produzca la interrupción. En este caso, existe lo que se denomina como robot *master* y robot *slave*.

Para poder realizar esta ampliación, se ha de conectar una salida del robot *master* con una entrada del robot *slave*.

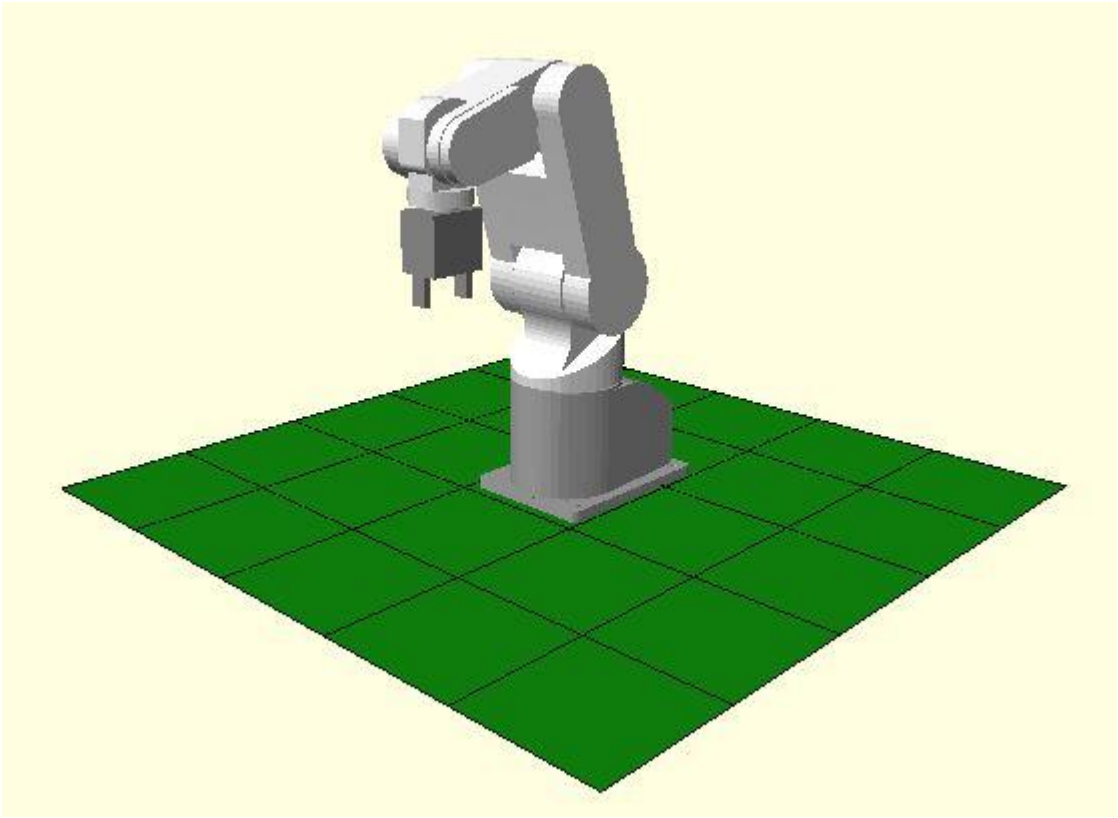
A6.3. Cuestiones finales

PRÁCTICA 6: CUESTIONES FINALES	**Fecha:**
Alumno:	**Grupo:**

1. Explica a qué crees que es debido que el programa no pueda compilar cuando escribimos la instrucción RETURN 0, para que vuelva al instante donde se produjo la interrupción. (2 puntos)
2. ¿Se puede crear otra alternativa para disminuir aún más el listado de posiciones? (1 punto)
2. ¿Existe alguna alternativa para realizar un bucle infinito que no sea WHILE 1? (1 punto)
3. ¿Qué otra señal, que no fuese de tiempo, podría hacer que se produjese una interrupción en esta misma aplicación? (1 punto)
4. ¿Por qué es necesario deshabilitar la interrupción en la subrutina Pegamento_Pieza2? (1 punto)
5. ¿Puede realizar un robot una interrupción sobre otro? Justifica tu respuesta (1 punto)
6. ¿En qué consiste el modelo de comunicación *master-slave*? (1 punto)
7. Redacta las conclusiones y conceptos aprendidos en la práctica 6. (2 puntos)

A7. PRÁCTICA 7

Automatización y Robótica Industrial



PRÁCTICA 7

A7.1. Desarrollo de la práctica

En esta última práctica, se mostrarán el conjunto de conceptos que se han podido aprender durante las prácticas anteriores mediante varios robots.

El primero de ellos se encargará de realizar una función de *Pick and Place* de cajas que aparecen por una cinta, las cuales vienen previamente de una aplicación de pintura. Las cajas que lleguen pintadas de azul y rojo se tomarán como correctas, y, por tanto, serán transportadas a otra cinta paralela al robot para que continúen en el proceso, mientras que las que vienen de gris se considerará que han tenido algún defecto en la pintura y deberán de ser desechadas en una caja. En esta primera aplicación, además de poner en práctica el *picking* de objetos, también tendrán presencia el uso de subrutinas, ya que la función de coger la caja de la cinta es común para todas las piezas.

El segundo robot, se encargará de poner adhesivo en las cajas de manera que el robot siga la forma de una S para aplicar cola por toda la superficie y a su vez se pongan en práctica las instrucciones de movimientos circulares, siendo necesario el uso de dos arcos para realizar la forma mencionada.

El tercer robot tendrá la función de colocar una pegatina sobre la cola introducida por el robot anterior. Cada caja, azul o roja, tendrá su correspondiente pegatina de manera que tendrán que ser recogidas de dos sitios independientes, pero la función de colocar la capa sobre la caja será la misma para todas las cajas, por tanto, podrá ser ejecutado mediante una subrutina.

Por último, encontraremos un cuarto robot que se encargará de realizar el paletizado de las piezas, colocando cada una de las cajas de un color en su correspondiente pallet, donde también encontraremos una subrutina para la acción de coger caja.

Finalmente, el resultado del *layout* debe ser parecido al de la fotografía que se muestra a continuación:

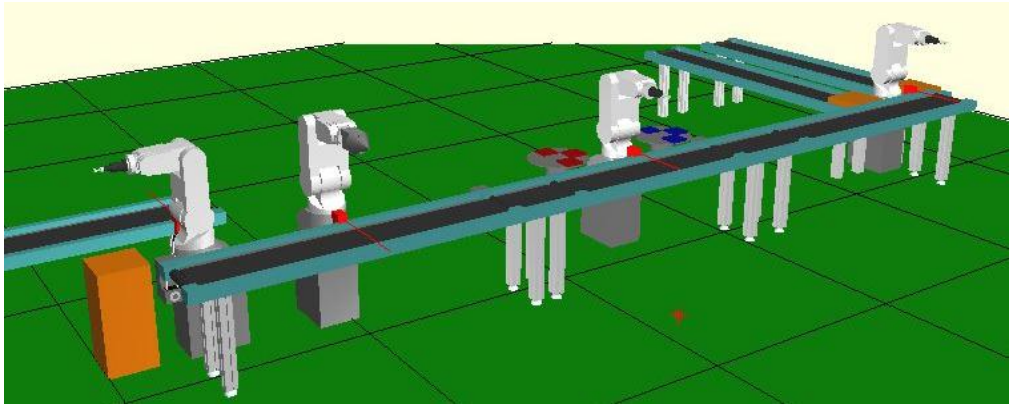


Figura A7.1. Ejemplo de layout para proyecto final (Fuente: Propia).

A continuación, se adjunta un link donde puede observarse la ejecución del proceso descrito en el enunciado:

https://www.youtube.com/watch?v=n_-JPteH1q0&feature=youtu.be

A7.1.1. Creación del *layout*

Para la posible realización del proceso que se muestra en el vídeo, a continuación, se adjuntan las tablas donde se indican las posiciones de los componentes presentes en el *layout*:

Tabla A7.1. Posiciones de los componentes del *layout* (Fuente: Propia).

	Posición			**Rotación**		
	X (mm)	**Y (mm)**	**Z (mm)**	**Roll (°)**	**Pitch (°)**	**Yaw (°)**
RV-2AJ	-300	-600	300	-90	0	0
RV-2AJ_1	-250	-50	350	0	0	0
RV-2AJ_2	-250	1500	300	0	0	0
RV-2AJ_3	-250	3500	300	0	0	0
ConveyorBelt	-480	-3800	0	90	0	0
ConveyorBelt_1	-480	-2200	0	90	0	0
ConveyorBelt_2	200	-800	0	90	0	0
ConveyorBelt_3	200	600	0	90	0	0
ConveyorBelt_4	200	1900	0	90	0	0
ConveyorBelt_5	-120	3400	-50	180	0	0
ConveyorBelt_6	-100	3900	-50	180	0	0
SensorColor	-500	-600	515	-180	90	0
SensorColor_1	-100	1500	515	0	90	0

SensorColor_2	-100	3550	515	0	90	0
SensorDistance	-500	-600	515	180	90	0
SensorDistance_1	-100	-50	515	0	90	0
TurnTable	-300	1175	340	0	0	0
TurnTable_1	-300	1825	340	0	0	0

Por otro lado, también se tiene las cajas que actúan como material, pegatina, caja de material defectuoso y los pallets. A continuación, se muestran las dimensiones y posiciones:

Tabla A7.4. Posiciones de las cajas del *layout* (Fuente: Propia).

	Dimensiones			**Posición**		
	X (mm)	**Y (mm)**	**Z (mm)**	**X (mm)**	**Y (mm)**	**Z (mm)**
Box (Caja mat. defectuoso)	200	200	450	-425	-1000	0
Box_1 (Pallet)	220	220	50	-370	3640	430
Box_2 (Pallet)	220	220	50	-370	3140	430
Box_3 (Material)	100	100	50	-	-	-
Box_4 (Pegatina)	90	90	10	-	-	-

Las posiciones del material son aleatorias, siendo situados centrados sobre la cinta de entrada. Las pegatinas deben ser situadas a los extremos de las mesas giratorias.

A7.1.2. Grabación de los puntos

Una vez se ha creado el *layout*, se procede a la grabación de los puntos para cada uno de los robots:

No	Position	Orientation	Comment
P3	300.0, -29.0, 195.0	0, 180,R,A	Dejada cajas incorrectas
P2	0.0, 338.7, 230.0	0, 180,R,A	Dejada cajas OK
P1	35.0,-335.0, 230.0	0, 180,R,A	Recogida cajas

Figura A7.2. Listado de puntos grabados para el robot RV-2AJ (Fuente: Propia).

No	Position	Orientation	Comment
P9	253.0, -5.0, 234.0	0, 180,R,A	Punto final S
P8	253.0, -55.0, 234.0	0, 180,R,A	Punto final arco2
P7	270.5, -75.0, 234.0	0, 180,R,A	Punto intermedio arco2
P6	288.0, -55.0, 234.0	0, 180,R,A	Punto inicial arco2
P5	288.0, -15.0, 234.0	0, 180,R,A	Punto final arco
P4	305.5, -5.0, 234.0	0, 180,R,A	Punto intermedio arco
P3	323.0, -15.0, 234.0	0, 180,R,A	Punto inicial arco
P2	323.0, -75.0, 234.0	0, 180,R,A	Punto inicio S
P1	230.0, 0.0, 340.0	0, 180,R,A	Posición inicial

Figura A7.3. Listado de puntos grabados para el robot RV-2AJ_1 (Fuente: Propia).

No	Position	Orientation	Comment
P3	45.0, 325.0, 205.0	0, 180,R,A	Coger capa azul
P2	45.0,-325.0, 205.0	0, 180,R,A	Coger capa roja
P1	285.0, -35.0, 230.0	0, 180,R,A	Poner capa

Figura A7.4. Listado de puntos grabados para el robot RV-2AJ_2 (Fuente: Propia).

No	Position	Orientation	Comment
P9	40.0,-200.0, 230.0	0, 180,R,A	P9 pallet azul
P8	-60.0,-200.0, 230.0	0, 180,R,A	P8 pallet azul
P7	40.0,-300.0, 230.0	0, 180,R,A	P7 pallet azul
P6	-60.0,-300.0, 230.0	0, 180,R,A	P6 pallet azul
P5	40.0, 200.0, 230.0	0, 180,R,A	P5 pallet rojo
P4	-60.0, 200.0, 230.0	0, 180,R,A	P4 pallet rojo
P3	40.0, 300.0, 230.0	0, 180,R,A	P3 pallet rojo
P2	-60.0, 300.0, 230.0	0, 180,R,A	P2 pallet rojo
P1	285.0, 10.0, 230.0	90, 180,R,A	Punto recogida cajas

Figura A7.5. Listado de puntos grabados para el robot RV-2AJ_3 (Fuente: Propia).

A7.1.3. Programación del proceso

Posteriormente a la grabación de los puntos, se procede a la programación para cada uno de los robots. A continuación, se adjunta el programa de movimientos creado para el primer robot (RV-2AJ):

```
'-------------------------------------------------------
'                        PRÁCTICA 7
'    Autor: Sandra Corchado
'    Fecha: Diciembre 2017
'-------------------------------------------------------
DEF INTE M1
M1=1
HOPEN 1
OVRD 100

MOV P1, -50
M_OUT(6)=1
M_OUT(7)=1
WHILE (M1=>1) AND (M1<=7)
IF M_IN(7)=1 THEN M_OUT(6)=0 AND M_OUT(7)=0 ELSE 20
SELECT M_IN(5) AND M_IN(6) AND M_IN(7)
CASE M_IN(5)=1
GOSUB *COGER_CAJA
MOV P2, -50
SPD 70
MVS P2
HOPEN 1
DLY 1
SPD 100
MVS P2, -50
MOV P1, -50
M1=M1+1
BREAK


CASE M_IN(6)=1
GOSUB *COGER_CAJA
MOV P2, -50
SPD 70
MVS P2
HOPEN 1
DLY 1
MVS P2, -50
SPD 100
MOV P1, -50
M1=M1+1
BREAK

DEFAULT
GOSUB *COGER_CAJA
MOV P3, -50
SPD 70
MVS P3
HOPEN 1
DLY 1
MVS P3, -50
SPD 100
MOV P1, -50
M1=M1+1
BREAK
END SELECT
WEND
END


*COGER_CAJA
MOV P1, -50
SPD 70
MVS P1
HCLOSE 1
DLY 1
MVS P1, -50
SPD 100
RETURN
```

El programa realizado para el segundo robot (RV-2AJ_1) es el siguiente:

```
10 '-------------------------------------------------------
11 '                        PRÁCTICA 7
12 '    Autor: Sandra Corchado
13 '    Fecha: Diciembre 2017
14 '-------------------------------------------------------
15 DEF INTE M1
16 M1=1
17 HOPEN 1
18 HCLOSE 1
19 OVRD 100

20 MOV P1
21 M_OUT(6)=1
22 WHILE (M1=>1 AND M1<=6)
23 IF M_IN(5)=1 THEN M_OUT(6)=0 ELSE 21
24 CNT 1, 50,50
25 MOV P2, -20
26 SPD 90
27 MVS P2
28 MVS P3
29 MVR P3,P4,P5
30 MVS P6
31 MVR P6,P7,P8
32 MVS P9
33 CNT 0
34 SPD 100
35 MOV P1
36 M_OUT(6)=1
37 DLY 1
38 M1=M1+1
39 WEND

40 M_OUT(6)=0
41 DLY 25
42 M_OUT(6)=1
43 END
```

Para el tercer robot (RV-2AJ_2), el programa realizado es el siguiente:

```
10 '-------------------------------------------------------
11 '                        PRÁCTICA 7
12 '    Autor: Sandra Corchado
13 '    Fecha: Diciembre 2017
14 '-------------------------------------------------------
15 DEF INTE M1
16 M1=1
17 HOPEN 1
18 OVRD 100
19 MOV P1, -50

20 M_OUT(6)=1
21 WHILE (M1=>1 AND M1<=6)
22 IF M_IN(5)=1 OR M_IN(6)=1 THEN M_OUT(6)=0 ELSE 20
```

```
23 SELECT M_IN(5) AND M_IN(6)
24 CASE M_IN(5)=1
25 MOV P2, -50
26 SPD 70
27 MVS P2
28 HCLOSE 1
29 DLY 1
30 MVS P2, -50
31 SPD 100
32 MOV P1,-50
33 GOSUB *PONER_PEGATINA
34 M_OUT(7)=0
35 M_OUT(7)=1
36 M_OUT(6)=1
37 DLY 1
38 M1=M1+1
39 BREAK

40 CASE M_IN(6)=1
41 MOV P3, -50
42 SPD 70
43 MVS P3
44 HCLOSE 1
45 DLY 1
46 MVS P3, -50
47 SPD 100
48 MOV P3,-50
49 GOSUB *PONER_PEGATINA
50 M_OUT(8)=0
51 M_OUT(8)=1
52 M_OUT(6)=1
53 DLY 1
54 M1=M1+1
55 BREAK
56 END SELECT
57 WEND
58 END

59 *PONER_PEGATINA
60 MOV P1, -50
61 MVS P1
62 HOPEN 1
63 DLY 1
64 MVS P1, -50
65 RETURN
```

Por último, el programa de movimientos realizado para el cuarto robot (RV-2AJ_3) es el siguiente:

```
10 '-------------------------------------------------------
11 '                        PRÁCTICA 7
12 '    Autor: Sandra Corchado
13 '    Fecha: Diciembre 2017
14 '-------------------------------------------------------
15 DEF PLT 1, P2, P3, P4, P5, 2, 2, 2
16 DEF PLT 2, P6, P7, P8, P9, 2, 2, 2
17 DEF INTE M1
18 DEF INTE M2
19 DEF INTE M3
20 M1=1
21 M2=1
22 M3=1
23 HOPEN 1
24 OVRD 100
25 MOV P1, -50
```

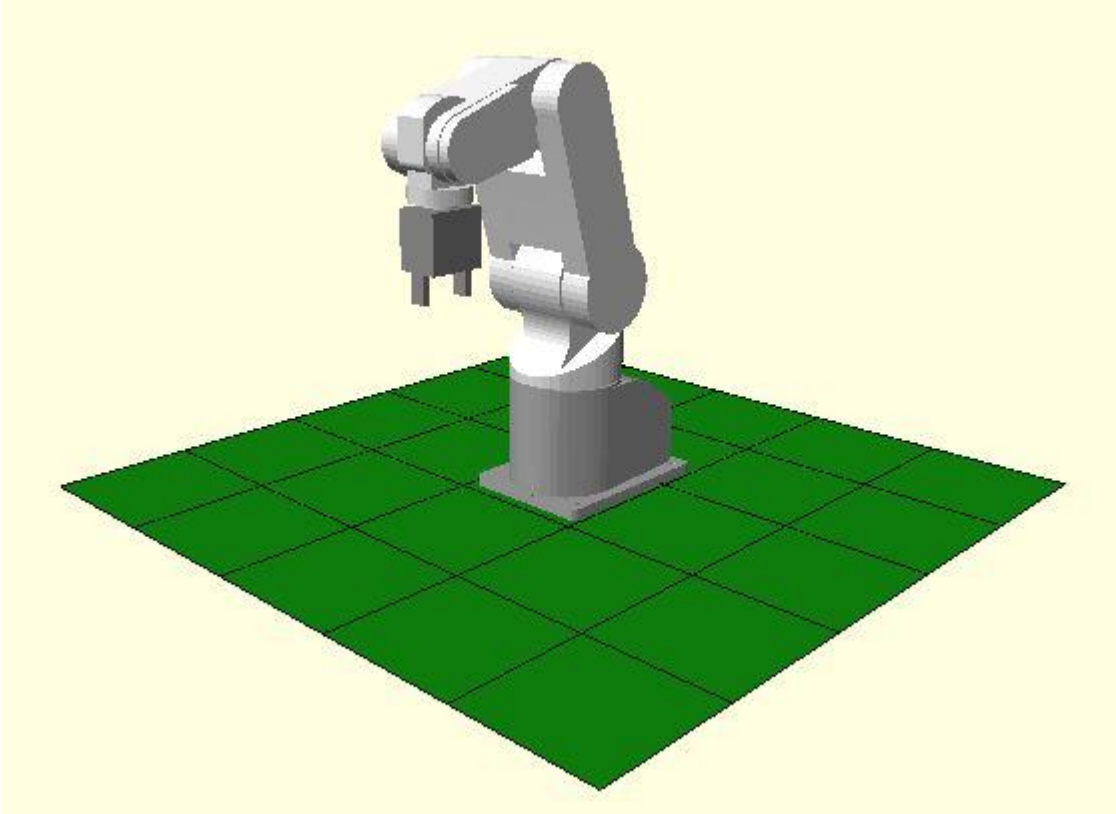
```
26 M_OUT(6)=1
27 WHILE (M3=>1 AND M3<=6)
28 IF M_IN(5)=1 OR M_IN(6)=1 THEN M_OUT(6)=0 ELSE 26
29 SELECT M_IN(5) AND M_IN(6)
30 CASE M_IN(5)=1
31 GOSUB *COGER_CAJA
32 P10=(PLT 1, M1)
33 MOV P10,-50
34 SPD 70
35 MVS P10
36 HOPEN 1
37 DLY 1
38 MVS P10,-50
39 SPD 100
40 MOV P1, -50
41 M1=M1+1
42 M_OUT(6)=1
43 DLY 1
44 M3=M3+1
45 BREAK


46 CASE M_IN(6)=1
47 GOSUB *COGER_CAJA
48 P11=(PLT 2, M2)
49 MOV P11,-50
50 SPD 70
51 MVS P11
52 HOPEN 1
53 DLY 1
54 MVS P11,-50
55 SPD 100
56 MOV P1, -50
57 M2=M2+1
58 M_OUT(6)=1
59 DLY 1
60 M3=M3+1
61 BREAK
62 END SELECT
63 WEND
64 END

65 *COGER_CAJA
66 MOV P1,-50
67 SPD 70
68 MVS P1
69 HCLOSE 1
70 DLY 1
71 MVS P1,-50
72 RETURN
```

A8. PROYECTO FINAL

Automatización y Robótica Industrial



PROYECTO FINAL

En este último apartado, se quiere llevar a cabo la creación de un proyecto de manera autónoma para que puedan ponerse en práctica todos los conceptos aprendidos en las prácticas anteriores.

El proceso a realizar será el correspondiente a la creación de las cápsulas de café que se utilizan en las cafeteras actuales de Nespresso. Existirán 4 tipos de cápsulas, por ejemplo, una puede ser la correspondiente al café descafeinado, otra para el café normal, otra café largo y una última de café con sabor a vainilla.

Se requiere la automatización de todo el proceso, desde la llegada de la capsula vacía hasta la colocación en cajas, siempre intentando crear el proceso óptimo, en cuanto a tiempo y precio.

A8.1. Posible resolución

- Las cápsulas vacías entran por una cinta (Entradas/Salidas).

- Un primer robot detecta la cápsula y coloca alternativamente una pegatina para diferenciar unas cápsulas de otras (Subrutinas).

- Segundo robot detecta el color de la cápsula según la pegatina y coloca cada una en su respectiva cinta (Pick and Place).

- De esta manera, se tendrán 4 cintas colocadas en paralelo, por lo que podemos colocar un tercer robot entre la primera y segunda cinta, y un cuarto robot entre la tercera y cuarta cinta. Cada una de estas cintas tendrá un sensor de presencia, por lo que el robot introducirá el café en la cápsula correspondiente según se activan las señales producidas por los sensores.

- A continuación, los seguirán los robots que coloquen la tapa a las cápsulas, en las que primeramente aplicarán adhesivo en el borde para después colocar la tapa. (Movimientos circulares y subrutinas).

- Seguidamente se encuentran los robots encargados de colocar las cápsulas en cajas. Recordad que debe haber una cantidad determinada de cápsulas en cada caja.

- Finalmente, encontraremos un robot al final de la línea que coloque las cajas de los 4 tipos en un pallet, colocando cada tipo en una fila determinada (Paletización).